

MATLAB 仿真应用精品丛书

# MATLAB R2016a

## 小波分析 22 个算法实现

方清城 编著

電子工業出版社

Publishing House of Electronics Industry

北京 • BEIJING

## 内 容 简 介

本书以 MATLAB R2016a 为平台编写,全面、系统地介绍了小波变换中的各种技术及应用。全书共 22 章,分别介绍了小波变换的基本概念、小波 MATLAB 工具箱、小波用于信号处理、小波用于图像处理、小波在实际工程中的应用、小波包算法应用、提升小波及其应用等内容。本书在编写过程中力求系统性、实用性与先进性相结合,理论与实践相交融,使读者可快速掌握 MATLAB 软件,同时利用 MATLAB 解决小波分析中的信号处理问题,达到学以致用目的。

本书可满足学习小波分析理论和 MATLAB 工程实践等不同层次读者的需要,包括小波分析爱好者,在校的本科生、研究生,相关培训机构的教师和学员,同时也可以作为工程技术人员的自学参考书。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。  
版权所有,侵权必究。

图书在版编目(CIP)数据

MATLAB R2016a 小波分析 22 个算法实现 / 方清城编著. —北京:电子工业出版社, 2018.1  
(MATLAB 仿真应用精品丛书)

ISBN 978-7-121-33391-0

M... 方... Matlab 软件—应用—小波理论 TP317 O174.22

中国版本图书馆 CIP 数据核字(2017)第 325736 号

策划编辑:陈韦凯

责任编辑:苏颖杰

印 刷:

装 订:

出版发行:电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本:787×1 092 1/16 印张:29 字数:740 千字

版 次:2018 年 1 月第 1 版

印 次:2018 年 1 月第 1 次印刷

印 数:2 000 册 定价:69.00 元

凡所购买电子工业出版社图书有缺损问题,请向购买书店调换。若书店售缺,请与本社发行部联系,联系及邮购电话:(010) 88254888, 88258888。

质量投诉请发邮件至 [zltz@phei.com.cn](mailto:zltz@phei.com.cn), 盗版侵权举报请发邮件至 [dbqq@phei.com.cn](mailto:dbqq@phei.com.cn)。

本书咨询联系方式:(010) 88254441; [chenwk@phei.com.cn](mailto:chenwk@phei.com.cn)。



# 前 言

MATLAB 对许多专门的领域都开发了功能强大的模块集和工具箱。一般来说，它们都是由特定领域的专家开发的，用户可以直接使用工具箱学习、应用和评估不同的方法而不需要自己编写代码。目前，MATLAB 已经把工具箱延伸到了科学研究和工程应用的诸多领域，如数据采集、数据库接口、概率统计、样条拟合、优化算法、偏微分方程求解、神经网络、小波分析、信号处理、图像处理、系统辨识、控制系统设计、LMI 控制、鲁棒控制、模型预测、模糊逻辑、金融分析、地图工具、非线性控制设计、实时快速原型及半物理仿真、嵌入式系统开发、定点仿真、DSP 与通信、电力系统仿真等。

MATLAB 的基本数据单位是矩阵，它的指令表达式与数学、工程中常用的形式十分相似，故用 MATLAB 来解算问题要比用 C、FORTRAN 等语言简洁得多。并且，MATLAB 也吸收了 Maple 等软件的优点，成为一个强大的数学软件。在新的版本中，MATLAB 还加入了对 C、FORTRAN、C++、Java 等语言的支持。

MATLAB R2016a 版本新增功能如下。

- MATLAB 实时编辑器：提供一种全新方式来创建、编辑和运行 MATLAB 代码，加快了探索性编程和分析的速度。
- App Designer：提供增强的设计环境和 UI 组件集，用于构建 MATLAB 应用程序的环境，简化了布置用户界面可视组件的过程。
- 深度学习：深度学习用于图像分类问题。
- Simulink Start Page：通过访问模板、最新模型以及精选示例可更快地开始或继续工作。
- SimEvents 新引擎：创建包含事件操作和新模块的离散事件模型和调度程序。
- 飞行仪器库：使用标准座舱仪器显示飞行条件。
- 通过访问模板、最近模型和精选示例可更快地开始或恢复工作的 Simulink Start Page。
- 自动设置求解器：可更快速地设置和仿真模型。
- 使用不同架构的目标器件的系统模型仿真，如 Xilinx 和 Altera SoC 架构。
- Simulink 单位：可在 Simulink、Stateflow 和 Simscape 组件的接口指定单位对其进行可视化处理并检查。
- 新增 Variant Source 和 VariantSink 模块，用于定义变量条件并使用生成代码中的编译器指令将其传播至连接的功能。

小波变换 (Wavelet Transform, WT) 是一种新的变换分析方法，它继承和发展了短时傅里叶 (Fourier) 变换局部化的思想，同时又克服了窗口大小不能随频率变化等缺点，能够提供—个随频率改变的“时间-频率”窗口，是进行信号时频分析和处理的理想工具。它的主要特点是通过变换能够充分突出问题某些方面的特征；能进行对时间 (空间) 频率的局部化分析，通过伸缩、平移运算对信号 (函数) 逐步进行多尺度细化，最终实现高频部

分时间细分，低频部分频率细分；能自动适应时频信号分析的要求，从而可聚焦到信号的任意细节，解决了 Fourier 变换困难的问题，成为继 Fourier 变换之后科学方法上的重大突破。

小波变换适用于信号的主要信息集中在低频段的情况。当感兴趣的频率成分位于中高频段时，如机械振动信号、语音信号等，由于小波变换在高频段的频谱窗口较宽，其小波系数中包含的频率成分过多，无法获取感兴趣的频率信号。利用小波包技术则可以将小波变换中停止分解的中高频段小波系数继续分解，使分解序列在整个时频域内都有较高的时频分辨率和相同的带宽。

与小波分解相比，小波包分解是一种更精细的分解方法，它不仅对图像的低频部分进行分解，而且对图像的高频部分进行分解。小波包对图像分解进行多分辨率分解是在小波函数对图像分解的基础上发展起来的，通过水平和垂直滤波，小波包变换将原始图像分为四个子带：水平和垂直方向上的低频子带、水平和垂直方向上的高频子带。相对于小波变换，小波包变换能够对图像中的高频部分进行分解，具有更强的适应性，因此更加适合于图像的各种处理。小波包分析属于线性时频分析法，它具有良好的时频定位特性以及对信号的自适应能力，因而能够对各种时变信号进行有效的分解。

考虑到小波变换域与 Fourier 变换域之间存在一定的转换关系，并且经典小波分析是在 Fourier 分析的基础上发展而来的，所以本书在附录中讲解了 Fourier 变换的基本理论及其在 MATLAB 中的实现，以便读者比较小波变换与 Fourier 变换的特点和处理问题的不同之处。

本书主要由方清城编写并统稿，参加编写的还有赵书兰、刘志为、栾颖、王宇华、吴茂、张德丰、李晓东、何正风、丁伟雄、李娅、辛焕平、杨文茵、顾艳春、邓奋发。

本书力求内容丰富、图文并茂、文字流畅，使之成为一本学习和使用 MATLAB 小波分析解决理论与工程应用问题方面有价值的参考书，但错误或疏漏之处在所难免，敬请读者批评指正。

编著者  
2017 年 9 月

# 目 录

第 1 章 提升算法的小波变换及其 MATLAB 实现 .....	1
1.1 MATLAB 实现提升方案的基本步骤 .....	1
1.2 MATLAB 小波工具箱函数 .....	2
1.2.1 添加原始或双重提升步骤函数 .....	3
1.2.2 一维提升小波变换函数 .....	3
1.2.3 提升方案信息函数 .....	3
1.2.4 转换滤波器为提升方案函数 .....	4
1.2.5 在四联滤波器上应用基本提升方案函数 .....	4
1.2.6 一维提升小波反变换函数 .....	5
1.2.7 显示提升方案函数 .....	5
1.2.8 提供常用小波的提升方案函数 .....	5
1.2.9 双正交尺度和小波函数 .....	6
1.2.10 提供小波的劳伦多项式函数 .....	7
1.2.11 二维提升小波变换函数 .....	7
1.2.12 提取或重构一维 LWT 小波系数函数 .....	7
1.2.13 劳伦矩阵类 LM 的构造器函数 .....	8
1.2.14 二维提升小波反变换函数 .....	8
1.2.15 提取或重构二维 LWT 小波系数函数 .....	9
1.2.16 劳伦多项类 LP 的构造器函数 .....	9
1.2.17 提供用于 LWT 的小波名信息函数 .....	9
1.3 MATLAB 提升小波函数应用 .....	10
第 2 章 基于小波变换的阈值去噪与图像压缩算法 .....	18
2.1 小波分析在图像去噪中的应用 .....	18
2.1.1 阈值处理函数的选取 .....	18
2.1.2 阈值的选取 .....	19
2.1.3 小波分析的去噪步骤 .....	19
2.1.4 小波分析去噪 MATLAB 例程 .....	20
2.2 基于小波分析的图像压缩 .....	25
2.2.1 图像小波分解的特点 .....	25
2.2.2 小波零树和 3 方向跨频带矢量的分类 .....	26



2.2.3	基于小波变换的图像局部压缩	27
2.2.4	小波变换用于图像压缩的一般方法	29
第 3 章	小波包算法分析与应用	39
3.1	小波包与信号去噪	39
3.1.1	基本原理	39
3.1.2	MATLAB 例程分析	40
3.2	小波包分析用于信号压缩	44
3.2.1	基本原理	44
3.2.2	MATLAB 例程分析	45
3.3	小波包与图像边缘检测	48
3.3.1	基本原理	48
3.3.2	MATLAB 例程分析	48
第 4 章	小波快速算法设计原理与实现	50
4.1	绪论	50
4.1.1	概述	50
4.1.2	傅里叶变换与小波变换的比较	51
4.1.3	小波分析与多辨分析的历史	52
4.2	从傅里叶变换到小波变换	53
4.2.1	傅里叶变换	54
4.2.2	短时傅里叶变换	54
4.2.3	小波变换	55
4.3	基于 MATLAB 的小波快速算法设计	58
4.3.1	小波快速算法设计原理与步骤	58
4.3.2	小波分解算法	59
4.3.3	对称小波分解算法	59
4.3.4	小波重构算法	60
4.3.5	对称小波重构算法	61
4.3.6	MATLAB 程序设计实现	61
第 5 章	利用小波变换对信号进行分析	73
5.1	信号压缩	73
5.1.1	信号压缩步骤	73
5.1.2	信号压缩实例	73
5.2	信号去噪	75
5.2.1	信号去噪步骤	75
5.2.2	信号去噪实例	76

5.3 信号分析与检测 .....	81
第 6 章 基于小波的间断点检测算法分析 .....	94
6.1 奇异性概念 .....	94
6.2 第一类间断点检测 .....	95
6.3 第二类间断点检测 .....	100
6.4 自相似检测 .....	103
6.5 信号的识别 .....	105
第 7 章 图像的小波分解算法与实现 .....	109
7.1 图像的小波分解算法 .....	109
7.2 小波变换系数分析 .....	111
7.3 实验结果与分析 .....	111
7.3.1 小波变换的图像压缩 .....	112
7.3.2 sym8 小波对图像进行分解 .....	114
7.3.3 小波系数分布理论分析 .....	120
第 8 章 提升小波变换的 MATLAB 实现 .....	128
8.1 MATLAB 一维提升小波变换 .....	128
8.1.1 一维信号压缩 wdcbm 函数应用 .....	128
8.1.2 一维信号压缩 ddencmp 函数应用 .....	129
8.1.3 信号去噪 .....	131
8.1.4 信号的提升分解 .....	133
8.1.5 信号的重构 .....	136
8.2 MATLAB 二维提升小波变换 .....	141
8.2.1 图像压缩 wdcbm2 函数应用 .....	141
8.2.2 图像压缩 ddencmp 函数应用 .....	142
8.2.3 图像去噪 .....	144
8.2.4 图像的提升分解 .....	146
8.2.5 图像的提升重构 .....	150
第 9 章 基于小波变换的回归估计与实现 .....	155
9.1 密度估计 .....	155
9.2 回归估计 .....	160
9.2.1 回归模型 .....	161
9.2.2 基于小波变换的回归估计 .....	161
9.2.3 小波变换实现回归估计 .....	163



第 10 章	信号的突变点检测算法分析与实现	167
10.1	信号的突变性与小波变换	167
10.2	信号的突变点检测原理	168
10.3	实验结果与分析	169
10.3.1	Daubechies 5 小波用于检测突变点	170
10.3.2	Daubechies 6 小波用于检测突变点	172
第 11 章	图像边缘检测算法分析与实现	175
11.1	多尺度边缘检测	175
11.2	快速多尺度边缘检测算法	177
11.3	实验结果与分析	178
第 12 章	二维小波变换的算法分析与实现	181
12.1	MATLAB 的图像处理	181
12.1.1	MATLAB 图像处理应用举例	181
12.1.2	图像处理基本操作	183
12.1.3	图像处理的高级应用	185
12.2	图像的小波分解和重构算法	187
12.2.1	二维小波变换及相应的快速算法	187
12.2.2	小波分解和重构 MATLAB 例程	192
第 13 章	函数的奇异性与故障信号检测分析	195
13.1	故障信号检测的理论分析	195
13.1.1	函数的奇异性	195
13.1.2	Lipschitz 指数分析	196
13.2	实验结果与分析	198
13.2.1	利用小波分析检测传感器故障	198
13.2.2	小波类型的选择对于检测突变信号的影响	202
13.3	小波类型选择	207
第 14 章	利用提升小波算法实现多分辨分析	209
14.1	小波分解与重构的多相位表示	210
14.2	Laurent 多项式 Euclidean 算法	211
14.3	改进的 Laurent 多项式 Euclidean 算法	212
14.4	多相位矩阵的因子分解	215
14.5	小波变换的提升实现的传统算法	219
14.6	小波变换的提升实现的简化算法	220
14.7	提升算法举例	221

14.8	整数小波变换	225
第 15 章	基于小波的阈值去噪方法分析	227
15.1	阈值去噪方法	227
15.2	阈值风险	228
15.3	实验结果与分析	229
15.3.1	利用小波分析对含噪正弦波进行去噪	230
15.3.2	小波分析对污染信号进行去噪处理	231
15.3.3	利用软、硬阈值去噪	233
第 16 章	连续与离散小波算法分析与实现	235
16.1	信号分解	235
16.1.1	信号的连续小波分解	235
16.1.2	信号的离散小波分解	242
16.2	信号重构	246
16.2.1	信号小波重构	246
16.2.2	小波函数应用实例	252
第 17 章	小波包在时频分析案例中的应用	261
17.1	小波包变换分析两个信号功率谱	261
17.2	调频信号的小波包分析	268
17.3	正弦信号的小波包分析	270
17.4	$\delta$ 信号的小波包分析	272
17.5	变频信号的小波包分析	274
第 18 章	小波在模态参数识别与化学中的应用	277
18.1	小波在化学中的应用	277
18.2	模态参数识别	281
18.2.1	模态时频辨识方法	281
18.2.2	小波脊线提取	282
18.2.3	改进 HHT 瞬时特征分析	282
18.2.4	模态参数识别的应用	282
第 19 章	小波变换图像测试分析	289
19.1	小波变换对图像压缩的步骤	289
19.2	实例说明	290
19.3	输出结果与分析	290
19.4	源程序	296



第 20 章 小波包分解与重构算法的应用	308
20.1 小波包基本理论	308
20.1.1 小波包理论分析	309
20.1.2 小波包的性质	310
20.1.3 小波包的空间分解	310
20.1.4 小波包算法	311
20.2 小波包函数用法	312
20.2.1 一维小波包的分解函数	312
20.2.2 一维小波包的重构函数	313
20.2.3 二维小波包的分解函数	314
20.2.4 二维小波包的重构函数	315
20.2.5 重新组合小波包函数	317
20.2.6 计算最佳树函数	319
20.2.7 小波包分析函数	321
20.2.8 更新小波包熵值函数	322
20.2.9 计算小波包熵函数	323
20.2.10 分割小波包函数	324
20.2.11 计算完整最佳小波包树函数	325
20.2.12 从小波包树中提取小波树函数	327
20.2.13 剪切小波包分解树函数	328
20.2.14 计算小波包系数函数	330
20.2.15 小波包分解系数的重构函数	331
第 21 章 多分辨分析及 Mallat 算法分析	336
21.1 小波分析的基本理论	336
21.2 连续小波变换	337
21.3 离散小波变换	338
21.4 多分辨分析及 Mallat 算法	338
21.5 一维正交多分辨分析及 Mallat 算法	338
21.6 紧支撑双正交小波基的构造	344
21.7 第二代小波变换	347
第 22 章 小波变换及其 MATLAB 例程分析	353
22.1 基于小波分析的图像平滑	353
22.1.1 小波图像平滑的基本原理	353
22.1.2 MATLAB 例程分析	353
22.2 基于小波变换数字图像水印研究	354
22.2.1 数字水印应具有的特点	355



22.2.2	数字水印的基本理论框架	356
22.2.3	数字水印技术需要解决的问题	357
22.2.4	一种基于小波变换的数字水印方法	357
22.2.5	MATLAB 例程分析	358
22.3	小波分析与图像增强	362
22.3.1	小波图像增强的基本方法	362
22.3.2	图像增强的 MATLAB 例程	363
22.4	小波分析与图像融合	368
22.4.1	小波图像融合的基本原理	368
22.4.2	MATLAB 例程分析	369
附录 A	MATLAB 的程序设计及绘图功能	372
A.1	MATLAB 程序设计原则	372
A.2	M 文件	372
A.3	MATLAB 的流程控制	375
A.4	MATLAB 的二维绘图	384
附录 B	Fourier 变换与 MATLAB 实现	397
B.1	复数形式的 Fourier 级数及其 MATLAB 应用	397
B.2	Fourier 变换的性质	401
附录 C	Fourier 变换分析与实现	416
C.1	Fourier 级数与 Fourier 变换	416
C.2	三角级数	417
C.3	以 $2\pi$ 为周期函数的 Fourier 级数	417
C.4	Fourier 变换	418
C.5	Fourier 变换及 MATLAB 实现	419
C.6	MATLAB 函数实现 Fourier 变换	420
C.7	连续时间信号 Fourier 变换的数值计算	422
C.8	信号的 Fourier 分解与合成 MATLAB 实现	423
附录 D	快速 Fourier 变换及其应用	429
D.1	快速 Fourier 变换及其 MATLAB 应用	429
D.2	运用 FFT 进行简单滤波	438
D.3	FFT 在工程分析中的应用	441
参考文献		449

# 第 1 章 提升算法的小波变换 及其 MATLAB 实现

## 1.1 MATLAB 实现提升方案的基本步骤

二维离散小波变换最有效的实现方法之一是采用 Mallat 算法，通过在图像的水平 and 垂直方向交替采用低通和高通滤波器实现。这种传统的基于卷积的离散小波变换计算量大、计算复杂高，对存储空间要求高，不利于硬件实现。提升小波的出现有效地解决了这一问题。提升算法相对于 Mallat 算法而言，是一种更为快速有效的小波变换实现方法，被誉为第二代小波变换。它不依赖于 Fourier 变换，继承了第一代小波的多分辨率特征，小波变换后的系数是整数，计算速度快，计算时无须额外的存储开销。Daubechies 已经证明，任何离散小波变换或具有有限长滤波器的两阶滤波变换都可以被分解成为一系列简单的提升步骤，所以能够用 Mallat 算法实现的小波，都可以用提升算法来实现。

提升算法给出了双正交小波简单而有效的构造方法，使用了基本的多项式插值来获取信号的高频分量，之后通过构建尺度函数来获取信号的低频分量。“提升”算法的基本思想是，将现有的小波滤波器分解成基本的构造模块，分步骤完成小波变换。

基于提升算法的小波变换称为第二代小波变换。它使我们能够用一种简单的方法去解释小波的基本理论，而第一代小波变换都可以找到等效的提升方案。提升方案把第一代小波变换过程分为以下三个阶段：分解（split）、预测（predict）和更新（update）。

### （1）分解。

将输入信号  $s_i$  分为 2 个较小的子集  $s_{i-1}$  和  $d_{i-1}$ ， $d_{i-1}$  也称小波子集。最简单的分解方法是输入信号  $s_i$  根据奇偶性分为 2 组，这种分裂所产生的小波称为懒小波（lazy wavelet），分解过程表示为  $F(s_i) = (s_{i-1}, d_{i-1})$ ，其中  $F(s_i)$  为分解过程。

### （2）预测。

在基于原始数据相关性的基础上，用偶数列  $s_{i-1}$  的预测值  $P(s_{i-1})$  去预测（或内插）奇数序列  $d_{i-1}$ ，即将滤波器  $P$  对偶信号作用以后作为奇信号的预测值，奇信号的实际值与预测值相减得到残差信号。实际中，虽然不可能从子集  $s_{i-1}$  中准确地预测子集  $d_{i-1}$ ，但是  $P(s_{i-1})$  有可能很接近  $d_{i-1}$ ，因此可以使用  $P(s_{i-1})$  和  $d_{i-1}$  的差来代替原来的  $d_{i-1}$ ，这样产生的  $d_{i-1}$  比原来的  $d_{i-1}$  包含更少的信息，于是得到  $d_{i-1} = d_{i-1} - P(s_{i-1})$ 。这里，已经可以用更小的子集  $s_{i-1}$  和小波子集  $d_{i-1}$  来代替原信号集  $s_i$ 。重复分解和预测过程，经  $n$  步以后，原信号集可用  $\{s_n, d_n, \dots, s_1, d_1\}$  来



表示。

(3) 更新。

为了使原信号集的某些全局特性在其子集  $s_{i-1}$  中继续保持, 必须进行更新。更新的思想是要找一个更好的子集  $s_{i-1}$ , 使得它保持原图的某一标量特性  $Q(x)$  (如均值、消失矩等不变), 即有  $Q(s_{i-1}) = Q(s_i)$ 。可能利用已经计算的小波子集  $d_{i-1}$  对  $s_{i-1}$  进行更新, 从而使得后者保持特性  $Q(x)$ , 即要构造一个算子  $U$  去更新  $s_{i-1}$ 。定义如下:

$$s_{i-1} = s_{i-1} + U(d_{i-1})$$

从上述可以知道, 提升方法可以实现原位运算, 即该方法不需要除了前级提升步骤的输出之外的数据, 这样, 在每个点都可以用新的数据流替换旧的数据流。当重复使用原位提升滤波器组时, 就获得了交织的小波变换系数。

## 1.2 MATLAB 小波工具箱函数

可以使用提升方法来设计新小波。这些提升方法允许“整数到整数”小波变换, 并使用不同长度的低通和高通分解滤波器进行小波变换。小波工具箱 3.0 版本中包括的新型提升函数主要有 5 组, 如表 1-1 所示。

表 1-1 提升函数

函数名称	函数名称	函数意义
提升方案函数	addlift	向提升方案中添加原始或双重提升步骤
	displs	显示提升方案
	lsinfo	提升方案信息
双正交四联滤波器	bswfun	计算并画出双正交“尺度和小波”函数
	filt2ls	将四联滤波器变换为提升方案
	liftfilt	在四联滤波器上应用基本提升方案
	ls2filt	将提升方案变换为四联滤波器
正交或双正交小波及 lazy 小波	liftwave	提升小波的提升方案
	wave2lp	提供小波的劳伦多项式
	wavenames	提供用于 LWT 的小波名
提升小波变换和反变换	lwt	一维提升小波变换
	lwt2	二维提升小波变换
	lwtcoef	提取或重构一维 LWT 小波系数
	lwtcoef2	提取或重构二维 LWT 小波系数
	ilwt	一维提升小波反变换
	ilwt2	二维提升小波反变换
劳伦多项式和矩阵	laurmat	劳伦矩阵类 LM 的构造器
	laurpoly	劳伦多项式类 LM 的构造器



### 1.2.1 添加原始或双重提升步骤函数

在 MATLAB 中实现添加原始或双重提升步骤的函数是 `addlift`，其调用格式有以下几种：

```
LSN=addlift(LS,ELS)
LSN=addlift(LS,ELS,'begin')
LSN=addlift(LS,ELS,'end')
```

格式 返回新的提升方案 LSN，它通过增加基本提升步骤 ELS 到提升方案 LS。

格式 预先考虑指定的基本提升步骤。

ELS 是单元阵列 {TYPEVAL, COEFS, MAX\_DEG} 或者结构 `struct('type',TYPEVAL,'value',LPVAL)`。其中，`LPVAL=laurpoly(COEFS,MAX_DE)`。

`LSN=addlift(LS,ELS,'end')` 等同于 `addlift(LS,ELS)`。

ELS 中有一串基本提升步骤，被存储在单元阵列或结构阵列中，然后这些提升步骤被增加到 LS。

### 1.2.2 一维提升小波变换函数

在 MATLAB 中实现一维提升小波变换的函数是 `lwt`，其调用格式有以下 5 种：

```
[CA,CD]=lwt(X,W)
X_InPlace= lwt(X,W)
lwt(X,W,LEVEL)
X_InPlace= lwt(X,W,LEVEL,'typeDEC',typeDEC)
[CA,CD]= lwt(X,W,LEVEL,'typeDEC',typeDEC)
```

`lwt` 对于给定的提升小波进行一维提升小波分解。该函数使用多项式算法。

格式 对向量 X 进行提升小波分解，计算出低频系数向量 cA 和低频向量 cD；W 是提升小波名。

格式 也是计算低频系数和高频系数。两个系数的存储方式为 `cA=X_InPlace(1:2:end)` 和 `cD=X_InPlace(2:2:end)`。

格式 进行 LEVEL 层计算提升小波分解。

格式 和 使用提升小波，当 `typeDEC='w'` 或 `typeDEC='wp'` 时，进行 LEVEL 层小波或小波包分解。

除了给定提升小波名外，还可以使用相关的提升方案 LS：`lwt(X,LS,...)` 来代替 `lwt(X,W,...)`。

参考函数：`ilwt`。

### 1.2.3 提升方案信息函数

在 MATLAB 中实现提升方案信息的函数是 `lsinfo`，其调用格式如下：

```
lsinfo
```

`lsinfo` 显示以下提升方案信息：提升方案 LS 是  $N \times 3$  单元阵列，前面的  $N-1$  行是基本的



提升步骤 (ELS), 最后一行是 LS 的归一化。

每个 ELS 具有的格式为 (type, coefficients, max\_degree)。其中, type 是 'p' (primal) 或者 'd' (dual); 系数 coefficients 是定义劳伦多项式 P 系数的向量; max\_degree 是 P 中单项的最高次数。

劳伦多项式 P 的形式为:

$$P(z) = C(1) * z^d + C(2) * z^{(d-1)} + \dots + C(m) * z^{(d-m+1)}$$

提升方案 LS 中, 对于 k 从 1 到 N-1, LS{k,:} 具有以下 ELS 的格式:

LS{k,1} 是提升类型 'p' (primal) 或 'd' (dual)。

LS{k,2} 是相应的提升滤波器。

LS{k,3} 是和 1 滤波器 LS{k,2} 相关的劳伦多项式的最高次数。

LS{N,1} 是 prima 的归一化 (实数)。

LS{N,2} 是 dual 的归一化 (实数)。

LS{N,3} 没有被使用。

通常, 归一化是指  $LS\{N,1\} * LS\{N,2\} = 1$ 。

例如, db1 相关的提升方案为

```
LS = { ...
      'd'      [ -1]      [0]
      'p'      [0.5000]    [0]
      [1.4142]  [0.7071]    []
    }
```

应用实例参见 displs 函数和 laurpoly 函数。

## 1.2.4 转换滤波器为提升方案函数

在 MATLAB 中实现转换滤波器为提升方案的函数是 filt2ls, 其调用格式如下:

```
LS=filt2ls(LoD,HiD,LoR,HiR)
```

该函数返回 4 个输入滤波器 LoD、HiD、LoR 和 HiR 的提升方案 LS。

## 1.2.5 在四联滤波器上应用基本提升方案函数

在 MATLAB 中实现在四联滤波器上应用基本提升方案的函数是 liftfilt, 其调用格式以下两种:

```
[LoDN,HiDN,LoRN,HiRN]=liftfilt(LoD,HiD,LoR,HiR,ELS)
[LoDN,HiDN,LoRN,HiRN]=liftfilt(LoD,HiD,LoR,HiR,ELS,TYPE,VALUE)
```

4 个滤波器 LoD、HiD、LoR 和 HiR 得到基本提升步骤 ELS, 格式 返回由 ELS 得到的 4 个 LoDN、HiDN、LoRN 和 HiRN。

ELS 的结构如下:

TYPE=ELS.type: 它包含基本提升步骤的类型。TYPE 可能的取值是 'p' (primal) 或 'd'



( dual )。

TYPE=ELS.value : 它包含基本提升步骤的劳伦多项式 T。如果 VALUE 是一个向量, 相关的劳伦多项式 T 就相当于 laurpoly(VALUE,0)。

另外, ELS 可以是一种尺度步骤, 这样, TYPE 为's'(scaling), VALUE 为尺度微分。

格式 返回同样的结果。

如果 TYPE='p', 则 HiD 和 LoR 保持不动; 如果 TYPE='d', 则 LoD 和 HiR 保持不动; 如果 TYPE='s', 则 4 个滤波器都保持不动。如果 ELS 是基本提升步骤的阵列, 则 liftfilt(...,ESL) 成功进行每一步。

### 1.2.6 一维提升小波反变换函数

在 MATLAB 中实现一维提升小波反变换的函数是 ilwt, 其调用格式有以下 6 种:

```
X=ilwt(AD_In_Place,W)
X=ilwt(CA,CD,W)
X=ilwt(AD_In_Place,W, LEVEL)
X=ilwt(CA,CD,W, LEVEL)
X=ilwt(AD_In_Place,W, LEVEL,'typeDEC',typeDEC)
X=ilwt(CA,CD,W, LEVEL,'typeDEC',typeDEC)
```

ilwt 使用指定的提升小波进行提升小波重构。

格式 使用提升小波重构得到的低频和低频系数向量 AD\_In\_Place, 计算重构向量 X; W 是提升小波名 (参见 liftwave 函数)。

格式 使用提升小波重构得到的低频系数向量 CA 和高频系数向量 CD, 计算重构向量 X。

格式 和 计算 LEVEL 层的提升小波重构。

格式 和 使用提升小波, 在 typeDEC='w'或 typeDEC='wp'时, 分别计算 LEVEL 层时的小波或小波包分解。

除了使用提升小波以外, 还可以使用相关的提升方案 LS :X=ilwt(...,LS,...)来代替 X= ilwt (... ,W,...)。

### 1.2.7 显示提升方案函数

在 MATLAB 中实现一维提升小波反变换的函数是 displs, 其调用格式如下:

```
S=displs(LS,FRM)
```

该函数返回描述提升方案 LS 的字符串。格式字符串 FRM 产生 S。

应用实例参见 addlift 函数。

### 1.2.8 提供常用小波的提升方案函数

在 MATLAB 中实现提供常用小波提升方案的函数是 liftwave, 其调用格式有以下两种:



```
LS=liftwave(WNAME)
LS=liftwave(WNAME,'Int2Int')
```

格式 返回由小波 WNAME 相关的提升方案。LS 是一个结构,而不是整数,被函数 lwt、ilwt 和 lwt2 等使用。

格式 进行整数到整数的小波变换。使用 'Int2Int' 产生 LS, 这样如果使用 [cA, cD]=lwt(X,LS) 或 Y=lwt(X,LS), 其中 X 是整数型向量, 那么结果 cA、cD 和 Y 都是整数型向量。如果忽略 'Int2Int', 那么 lwt 将产生实数向量。

WNAME 的有效值如表 1-2 所示。

表 1-2 WNAME 的有效值

WNAME 值	说 明
'lazy'	'lazy'小波指的是二次小波,并不是真实的数学小波
'Haar'	和'db1'、'bior1.1'、'cdf1.1'一样
'db1', 'db2', 'db3', 'db4', 'db5', 'db6', 'db7', 'db8' 'sym2', 'sym3', 'sym4', 'sym5', 'sym6', 'sym7', 'sym8'	'db2'和'sym2'一样, 'db3'和 'sym4'一样
Cohen-Daubechies-Feauveau 小波 'cdf1.1', 'cdf1.3', 'cdf1.5' 'cdf3.1', 'cdf3.3', 'cdf3.5', 'cdf5.1', 'cdf5.3', 'cdf5.5', 'cdf2.2', 'cdf2.4', 'cdf2.6' 'cdf4.2', 'cdf4.4', 'cdf4.6' 'cdf6.2', 'cdf6.4', 'cdf6.6'	'cdfX.Y'和'biorX.Y'一样,除了' bior 4.4'和'bior5.5'
'biorX.Y'	参见 waveinfo 函数
'rbioX.Y'	'biorX.Y'的逆小波,参见 waveinfo 函数
'bs3'	和'cdf4.2'一样
'rbs3'	'bs3'的逆小波
'9.7'	和'cdf4.4'一样
'r9.7'	'9.7'的逆小波

### 1.2.9 双正交尺度和小波函数

在 MATLAB 中实现双正交尺度和小波的函数是 bswfun, 其调用格式有以下 3 种:

```
[PHIS,PSIS,PHIA, PSIA,XVAL]=bswfun(LoD,HiD,LoR,HiR)
[PHIS,PSIS,PHIA, PSIA,XVAL]=bswfun(LoD,HiD,LoR,HiR,ITER)
[PHIS,PSIS,PHIA, PSIA,XVAL]=bswfun(LoD,HiD,LoR,HiR,'plot')
```

该函数使用级联算法。格式 返回两对滤波器(LoD,HiD)与 (LoR,HiR) 的两对尺度和小波函数(PHIA,PSIA)和(PHIS,PSIS)在网格 XVAL 上的近似值。

格式 进行 ITER 次反复调用, 计算两对尺度和小波函数。

格式 计算并画出这些函数。



### 1.2.10 提供小波的劳伦多项式函数

在 MATLAB 中实现提供小波的劳伦多项式的函数是 `wave2lp`，其调用格式如下：

```
[Hs,Gs,Ha,Ga]= wave2lp(W)
```

该函数返回小波  $W$  相关的 4 个劳伦多项式。 $H$ -多项式 ( $G$ -多项式) 是低通多项式。对于正交小波， $H_s=H_a$  且  $G_s=G_a$ 。

### 1.2.11 二维提升小波变换函数

在 MATLAB 中实现二维提升小波变换的函数是 `lwt2`，其调用格式有以下 5 种：

```
[CA,CH,CV,CD]=lwt2(X,W)
X_InPlace= lwt2(X,LS)
lwt2(X,W,LEVEL)
X_InPlace= lwt2(X,W,LEVEL,'typeDEC',typeDEC)
[CA,CD]= lwt2(X,W,LEVEL,'typeDEC',typeDEC)
```

`lwt2` 对于给定的提升小波进行二维提升小波分解。该函数使用多项式算法。

格式 对向量  $X$  进行提升小波分解，计算出低频系数向量  $CA$  和低频向量  $CH$ 、 $CV$  和  $CD$ 。 $W$  是提升小波名。

格式 也是计算低频系数和高频系数。两个系数的存储方式为：

```
CA=X_InPlace(1:2:end, 1:2:end)
```

```
CH=X_InPlace(2:2:end, 1:2:end)
```

```
CV=X_InPlace(1:2:end, 2:2:end)
```

```
CD=X_InPlace(2:2:end, 2:2:end)
```

格式 进行尺度为  $LEVEL$  时的提升小波分解。

格式 和 使用提升小波，当 `typeDEC='w'` 或 `typeDEC='wp'` 时进行  $LEVEL$  层小波或小波包分解。

除了给定提升小波名以外，还可以使用相关提升方案  $LS$ ：`lwt2(X,LS,...)` 来代替 `lwt2(X,W,...)`。

参考函数：`ilwt2`。

### 1.2.12 提取或重构一维 LWT 小波系数函数

在 MATLAB 中实现提取或重构一维 LWT 小波系数的函数是 `lwtcoef`，其调用格式有以下两种：

```
Y=lwtcoef(TYPE,XDEC,LS,LEVEL,LEVEXT)
```

```
Y=lwtcoef(TYPE,XDEC,W,LEVEL,LEVEXT)
```

格式 返回由  $XDEC$  提取的尺度为  $LEVELXT$  的系数或重构系数。 $XDEC$  是由提升方案  $LS$  得到的尺度为  $LEVEL$  层的分解。





TYPE 的有效值如表 1-3 所示。

表 1-3 TYPE 的有效值

TYPE 值	说 明
'a'	低频
'd'	高频
'ca'	低频系数
'cd'	高频系数

格式 使用提升小波 W，返回同样的输出。

### 1.2.13 劳伦矩阵类 LM 的构造器函数

在 MATLAB 中实现劳伦矩阵类 LM 的构造器的函数是 `laurmat`，其调用格式如下：

```
M=laurmat(V)
```

该函数返回劳伦矩阵对象 M，V 可以是劳伦多项式的单元阵列（最多二维）或普通的矩阵。

### 1.2.14 二维提升小波反变换函数

在 MATLAB 中实现二维提升小波反变换的函数是 `ilwt2`，其调用格式有以下 6 种：

```
X=ilwt2(AD_In_Place,W)
X=ilwt2(CA,CH,CV,CD,W)
X=ilwt2(AD_In_Place,W, LEVEL)
X=ilwt2(CA,CH,CV,CD,W, LEVEL)
X=ilwt2(AD_In_Place,W, LEVEL,'typeDEC',typeDEC)
X=ilwt2(CA,CH,CV,CD,W, LEVEL,'typeDEC',typeDEC)
```

`ilwt2` 使用指定的提升小波进行提升二维小波重构。

格式 使用提升小波重构得到的低频和低频系数矩阵 AD\_In\_Place，计算重构矩阵 X。W 是提升小波名（参见 `liftwave` 函数）。

格式 使用提升小波重构得到的低频系数矩阵 CA 和高频系数向量 CH、CV 和 CD，计算重构矩阵 X。

格式 和 计算尺度 LEVEL 时的提升小波重构。

格式 和 使用提升小波，在 `typeDEC='w'` 或 `typeDEC='wp'` 时，分别计算尺度 LEVEL 时的小波或小波包分解。

除了使用提升小波以外，还可以使用相关的提升方案 LS：`X=ilwt2(...,LS,...)` 来代替 `X=ilwt2(...,W,...)`。

参考函数：`lwt2`。

### 1.2.15 提取或重构二维 LWT 小波系数函数

在 MATLAB 中实现提取或重构二维 LWT 小波系数的函数是 `lwtcoef2`，其调用格式有以下两种：

```
Y=lwtcoef2(TYPE,XDEC,LS,LEVEL,LEVEXT)
Y=lwtcoef2(TYPE,XDEC,W,LEVEL,LEVEXT)
```

**格式** 返回由 XDEC 提取的尺度为 LEVELXT 的系数或重构系数。XDEC 是由提升方案 LS 得到的尺度 LEVEL 的分解。

TYPE 的有效值如表 1-4 所示。

表 1-4 TYPE 的有效值

TYPE 值	说 明
'a'	低频
'h'	水平高频
'v'	垂直高频
'd'	对角高频
'ca'	低频系数
'ch'	水平高频系数
'cv'	垂直高频系数
'cd'	对角高频系数

**格式** 使用提升小波 W，返回同样的输出。

### 1.2.16 劳伦多项类 LP 的构造器函数

在 MATLAB 中实现劳伦多项类 LP 的构造器的函数是 `laurpoly`，其调用格式有以下 3 种：

```
P=laurpoly(C,d)
P=laurpoly(C,'dmin',d)
P=laurpoly(C,'dmax',d)
```

该函数返回劳伦多项式对象。C 是向量，它的元素是多项式 P 的系数；d 是 P 的最高次数。

如果 m 是向量 C 的长度，则 P 代表以下的劳伦多项式：

$$P(z)=C(1)*z^d+C(2)*z^{(d-1)}+...+C(m)*z^{(d-m+1)}$$

**格式** 指定的是 P 的最低次数，而不是最高次数。相应的输出 P 代表以下劳伦多项式：

$$P(z)=C(1)*z^{(d+m-1)}+...+C(m-1)*z^{(d+1)}+...+C(m)*z^d$$

`P=laurpoly(C,'dmax',d)` 等同于 `P=laurpoly(C,d)`。

### 1.2.17 提供用于 LWT 的小波名信息函数

在 MATLAB 中实现提供用于 LWT 的小波名信息的函数是 `wavenames`，其调用格式如下：



```
W=wavenames(T)
```

该函数返回包含小波类型 T 的单元数组，T 的有效值包括：

'all'——所有小波；

'lazy'——'lazy'小波；

'orth'——正交小波；

'bior'——双正交小波。

W=wavenames 等同于 W=wavenames('all')。

## 1.3 MATLAB 提升小波函数应用

【例 1-1】lwtcoef2 函数应用举例。

```
% 使用 Haar 小波，得到相应的提升方案
lshaar=liftwave('haar');
% 添加 ELS 到提升方案
els={'p',[-0.125 0.125],0}
lsnew=addlift(lshaar,els);
% 对于简单图像，尺度为 2 进行 LWT
x=reshape(1:16,4,4);
xDec=lwt2(x,lsnew,2)
% 提取第一层的低频系数
ca1=lwtcoef2('ca',xDec,lsnew,2,1)
% 重构低频和高频
a1=lwtcoef2('a',xDec,lsnew,2,1)
a2=lwtcoef2('a',xDec,lsnew,2,2)
h1=lwtcoef2('h',xDec,lsnew,2,1)
v1=lwtcoef2('v',xDec,lsnew,2,1)
d1=lwtcoef2('d',xDec,lsnew,2,1)
h2=lwtcoef2('h',xDec,lsnew,2,2)
v2=lwtcoef2('v',xDec,lsnew,2,2)
d2=lwtcoef2('d',xDec,lsnew,2,2)
% 检查重构效果
err=max(max(abs(x-a2-h2-v2-d2-h1-v1-d1)))
```

程序运行结果如下：

```
els =
    'p'    [1x2 double]    [0]
xDec =
    27.4375    4.0000    17.0000    4.0000
    1.0000     0        1.0000     0
    4.2500    4.0000    0.0000    4.0000
    1.0000     0        1.0000     0
```



```

cal =
    5.7500    22.7500
   10.0000    27.0000

a1 =
    2.8750    2.8750   11.3750   11.3750
    2.8750    2.8750   11.3750   11.3750
    5.0000    5.0000   13.5000   13.5000
    5.0000    5.0000   13.5000   13.5000

a2 =
    6.8594    6.8594    6.8594    6.8594
    6.8594    6.8594    6.8594    6.8594
    6.8594    6.8594    6.8594    6.8594
    6.8594    6.8594    6.8594    6.8594

h1 =
   -0.3750   -0.3750   -0.3750   -0.3750
    0.6250    0.6250    0.6250    0.6250
   -0.5000   -0.5000   -0.5000   -0.5000
    0.5000    0.5000    0.5000    0.5000

v1 =
   -1.5000    2.5000   -2.0000    2.0000
   -1.5000    2.5000   -2.0000    2.0000
   -1.5000    2.5000   -2.0000    2.0000
   -1.5000    2.5000   -2.0000    2.0000

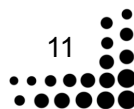
d1 =
         0         0         0         0
         0         0         0         0
         0         0         0         0
    0         0         0         0

h2 =
   -0.7969   -0.7969   -0.7969   -0.7969
   -0.7969   -0.7969   -0.7969   -0.7969
    1.3281    1.3281    1.3281    1.3281
    1.3281    1.3281    1.3281    1.3281

v2 =
   -3.1875   -3.1875    5.3125    5.3125
   -3.1875   -3.1875    5.3125    5.3125
   -3.1875   -3.1875    5.3125    5.3125
   -3.1875   -3.1875    5.3125    5.3125

d2 =
1.0e-015 *
    0.2498    0.2498   -0.4163   -0.4163
    0.2498    0.2498   -0.4163   -0.4163
   -0.4163   -0.4163    0.6939    0.6939

```





```

-0.4163    -0.4163    0.6939    0.6939
err =
3.5527e-015

```

应用实例参见 ilwt2 和 lwt2 函数。

【例 1-2】laurmat 函数应用举例。

```

M1=laurmat(eye(2,2))
Z=laurpoly(1,1);
M2=laurmat({1 Z;0 1})
% 计算劳伦多项式
P=M1*M2
d=det(P)

```

程序运行结果如下：

```

M1 =
| 1      0      |
|              |
|              |
| 0      1      |

M2 =
| 1      z^(+1)  |
|              |
|              |
| 0      1      |

P =
| 1      z^(+1)  |
|              |
|              |
| 0      1      |

d(z) = 1

```

应用实例参见 laurpoly 函数。

【例 1-3】liftfilt 函数应用举例。

```

% 得到 Haar 滤波器
[LoD,HiD,LoR,HiR]=wfilters('haar')
% 提升 Haar 滤波器
twoels(1)=struct('type','p','value',laurpoly([0.125 -0.125],0));
twoels(2)=struct('type','p','value',laurpoly([0.125 -0.125],1));
[LoDN,HiDN,LoRN,HiRN]=liftfilt(LoD,HiD,LoR,HiR,twoels);
% 得到双正交小波 bior1.3
[LoDB,HiDB,LoRB,HiRB]=wfilters('bior1.3');
somewavelet=isequal([LoDB,HiDB,LoRB,HiRB],[LoDN,-HiDN,LoRN,HiRN])

```



程序运行结果如下：

```
somewavelet =  
1
```

应用实例参见 laurpoly 函数。

【例 1-4】lwtcoef 函数应用举例。

```
% 使用 Haar 小波，得到相应的提升方案  
lshaar=liftwave('haar');  
% 添加 ELS 到提升方案  
els={'p',[ -0.125 0.125],0}  
lsnew=addlift(lshaar,els);  
% 对于简单信号，进行 2 层 LWT  
x=1:8;  
xDec=lwt(x,lsnew,2)  
% 提取尺度为 1 时的低频系数  
ca1=lwtcoef('ca',xDec,lsnew,2,1)  
% 重构低频和高频  
a1=lwtcoef('a',xDec,lsnew,2,1)  
a2=lwtcoef('a',xDec,lsnew,2,2)  
d1=lwtcoef('d',xDec,lsnew,2,1)  
d2=lwtcoef('d',xDec,lsnew,2,2)  
% 检查重构效果  
err=max(abs(x-a2-d2-d1))
```

程序运行结果如下：

```
els =  
    'p'    [1x2 double]    [0]  
xDec =  
    4.3438    0.7071    2.1250    0.7071    13.0313    0.7071    2.0000    0.7071  
ca1 =  
    1.9445    4.9497    7.7782    10.6066  
a1 =  
    1.3750    1.3750    3.5000    3.5000    5.5000    5.5000    7.5000    7.5000  
a2 =  
    2.1719    2.1719    2.1719    2.1719    6.5156    6.5156    6.5156    6.5156  
d1 =  
   -0.3750    0.6250   -0.5000    0.5000   -0.5000    0.5000   -0.5000    0.5000  
d2 =  
   -0.7969   -0.7969    1.3281    1.3281   -1.0156   -1.0156    0.9844    0.9844  
err =  
9.9920e-016
```

应用实例参见 ilwt 和 lwtt 函数。

【例 1-5】laurpolyt 函数应用举例。



```
% 定义劳伦多项式
P=laurpoly([1:3],2);
P=laurpoly([1:3],'dmax',2)
P=laurpoly([1:3],'dmin',2)
% 计算劳伦多项式
Z=laurpoly(1,1)
Q=Z*P
```

程序运行结果如下：

```
P(z) = + z^(+2) + 2*z^(+1) + 3
P(z) = + z^(+4) + 2*z^(+3) + 3*z^(+2)
Z(z) = z^(+1)
Q(z) = + z^(+5) + 2*z^(+4) + 3*z^(+3)
```

应用实例参见 laurmatt 函数。

**【例 1-6】** wave2lp 函数应用举例。

```
% 得到"lazy"小波相关的劳伦多项式
[Hs,Gs,Ha,Ga]= wave2lp('lazy')
```

程序运行结果如下：

```
Hs(z) = 1
Gs(z) = z^(-1)
Ha(z) = 1
Ga(z) = z^(-1)

% 得到 db1 小波相关的劳伦多项式
[Hs,Gs,Ha,Ga]= wave2lp('db1')
```

程序运行结果如下：

```
Hs(z) = + 0.7071 + 0.7071*z^(-1)
Gs(z) = - 0.7071 + 0.7071*z^(-1)
Ha(z) = + 0.7071 + 0.7071*z^(-1)
Ga(z) = - 0.7071 + 0.7071*z^(-1)

% 得到 bior1.3 小波相关的劳伦多项式 bior1.3
[Hs,Gs,Ha,Ga]= wave2lp('bior1.3')
```

程序运行结果如下：

```
Hs(z) = + 0.7071 + 0.7071*z^(-1)
Gs(z) = ...
+ 0.08839*z^(+2) + 0.08839*z^(+1) - 0.7071 + 0.7071*z^(-1) - 0.08839*z^(-2) ...
- 0.08839*z^(-3)
Ha(z) = ...
- 0.08839*z^(+2) + 0.08839*z^(+1) + 0.7071 + 0.7071*z^(-1) + 0.08839*z^(-2) ...
- 0.08839*z^(-3)
```



$$Ga(z) = -0.7071 + 0.7071 * z^{(-1)}$$

【例 1-7】filt2ls 函数应用举例。

```
[LoD,HiD,LoR,HiR]=wfilters('db2')
LS=filt2ls(LoD,HiD,LoR,HiR);
displs(LS);
LSref=liftwave('db2');
displs(LSref);
```

程序运行结果如下：

```
LoD =
    -0.1294    0.2241    0.8365    0.4830
HiD =
    -0.4830    0.8365   -0.2241   -0.1294
LoR =
    0.4830    0.8365    0.2241   -0.1294
HiR =
    -0.1294   -0.2241    0.8365   -0.4830
LS = {...
    'd'          [-1.73205081]          [0]
    'p'          [-0.06698730  0.43301270] [1]
    'd'          [ 1.00000000]          [-1]
    [ 1.93185165] [ 0.51763809]          []
};
LSref = {...
    'd'          [-1.73205081]          [0]
    'p'          [-0.06698730  0.43301270] [1]
    'd'          [ 1.00000000]          [-1]
    [ 1.93185165] [ 0.51763809]          []
};
```

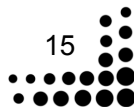
应用实例参见 ls2filt 和 isinfo 函数。

【例 1-8】liftwave 函数应用举例。

```
% 由 db2 小波得到相应的提升方案
lsdb2=liftwave('db2');
% 显示得到的提升方案
displs(lsdb2);
```

程序运行结果如下：

```
lsdb2 = {...
    'd'          [-1.73205081]          [0]
    'p'          [-0.06698730  0.43301270] [1]
    'd'          [ 1.00000000]          [-1]
    [ 1.93185165] [ 0.51763809]          []
};
```







应用实例参见 laurpoly 函数。

【例 1-9】displs 函数应用举例。

% 使用 Cohen-Daubechies-Feauveau 小波计算相应的提升方案

```
lscdf=liftwave('cdf3.1');
```

% 显示得到的提升方案

```
displs(lscdf);
```

% 将提升方案转换为双正交滤波器 quadruplet

```
[LoD,HiD,LoR,HiR]=ls2filt(lscdf);
```

% 显示两种尺度和小波函数

```
bswfun(LoD,HiD,LoR,HiR,'plot');
```

程序运行结果如图 1-1 所示。

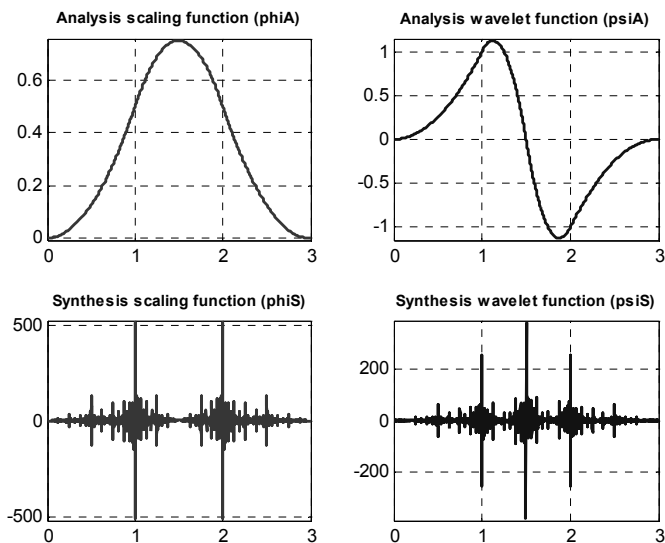


图 1-1 例程的运行结果

应用实例参见 wavefun 函数。

【例 1-10】addlift 函数应用举例。

% 由 Haar 小波，得到相应的提升方案

```
lshaar=liftwave('haar');
```

% 显示得到的提升方案

```
displs(lshaar);
```

% 添加到 ELS 到提升方案

```
els={'p',[-0.125 0.125],0};
```

```
lsnew=addlift(lshaar,els);
```

```
displs(lsnew);
```

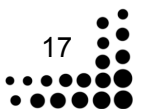
程序运行结果如下：

```
lshaar = {...
```



```
'd'          [-1.00000000]          [0]
'p'          [ 0.50000000]          [0]
[ 1.41421356] [ 0.70710678]          []
};
lsnew = {...
'd'          [-1.00000000]          [0]
'p'          [ 0.50000000]          [0]
'p'          [-0.12500000 0.12500000][0]
[ 1.41421356] [ 0.70710678]          []
};
```

参考函数：liftfilt。



## 第2章 基于小波变换的阈值去噪与图像压缩算法

### 2.1 小波分析在图像去噪中的应用

噪声可以理解为妨碍人的视觉器官或系统传感器对所接收图像源进行理解或分析的各种因素。一般噪声是不可预测的随机信号，它只能用概率统计的方法去认识。噪声对图像处理十分重要，它影响图像处理的输入、采集、处理的各个环节以及输出结果的全过程。特别是图像的输入、采集的噪声是个十分关键的问题，若输入伴有较大噪声，则必然影响处理全过程及输出结果。因此，一个好的图像处理系统，不论是模拟处理还是计算机处理，无不把减少最前级的噪声作为主攻目标。去噪已成为图像处理中极其重要的步骤。

#### 2.1.1 阈值处理函数的选取

Donoh 将阈值处理函数分为软阈值和硬阈值，如图 2-1 所示为两种阈值处理函数示意图。设  $w$  是小波系数的大小， $w_\lambda$  是施加阈值后的小波系数大小， $\lambda$  是阈值。

(1) 硬阈值 (hard thresholding)

当小波系数的绝对值小于给定阈值时，令其为 0，而大于阈值时，保持其不变，即

$$w_\lambda = \begin{cases} w, & |w| \geq \lambda \\ 0, & |w| < \lambda \end{cases}$$

(2) 软阈值 (soft thresholding)

当小波系数的绝对值小于给定阈值时，令其为 0，大于阈值时，令其都减去阈值，即

$$w_\lambda = \begin{cases} \text{sign}(w)(|w| - \lambda), & |w| \geq \lambda \\ 0, & |w| < \lambda \end{cases}$$

硬阈值函数在  $|w| = \lambda$  处是不连续的，容易造成去噪后的图像在奇异点附近出现明显的 Pseudo-Gibbs 现象。因此，本系统选用软阈值函数作为阈值处理函数。

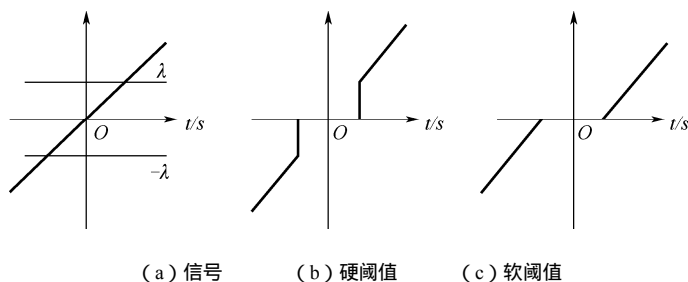


图 2-1 两种阈值处理函数示意图

### 2.1.2 阈值的选取

阈值的选择是离散小波去噪中最关键的一步。在去噪过程中,小波阈值  $\lambda$  起到了决定性作用:如果阈值太小,则施加阈值后小波系数将包含过多的噪声分量,达不到去噪的效果;反之,如果阈值太大,则去除了有用的成分,造成失真。所以对阈值的估计非常重要。

目前,所使用的阈值包括全局阈值和局部适应阈值,各种各样的阈值公式也曾出不穷。考虑到算法实现的复杂程度以及去噪的效果,大多数系统采用了 Donoho 和 Johnstone 的统一阈值  $\delta = \sigma\sqrt{2\lg N}$ 。其中,  $\sigma$  为噪声标准方差;  $N$  为信号的尺寸或长度。这是在正态高斯噪声模型下,针对多维独立正态变量联合分布,在维数趋向无穷时的研究得出的结论,即大于该阈值的系数含有噪声信号的概率趋于零。这个阈值由于同信号的尺寸对数的平方根成正比,所以当  $N$  较大时,阈值趋向于将所有小波系数置零。此时小波滤噪器退化为低通滤波器。

然而,在实际环境中,图像中的噪声标准方差是不能知道的,因此在选取阈值时,要对用估计方法来确定噪声标准方差。其中较常用的估算方法多采用如下公式:

$$\sigma = \frac{\text{median} \left[ |d_j(k)| \right]}{0.6745}$$

其中,  $j$  是小波分解尺度; median 是 MATLAB 中求中值得运算命令。

这种计算方法比较复杂,去噪效果也不能令人满意。本系统采用一种简单有效的阈值估计方法,在图像进行正交小波分解的第一级,取小波系数中的 HH 部分,根据它的标准方差  $\sigma$  的估计值  $\bar{\sigma}$ ,再利用 Donoho 和 Johnstone 统一阈值计算出阈值。

### 2.1.3 小波分析的去噪步骤

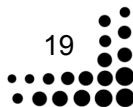
对二维图像信号的去噪方法同样适用于一维信号,尤其是对于几何图像更适合。二维模型可以表述为

$$s(i, j) = f(i, j) + \delta e(i, j) \quad i, j = 0, 1, \dots, m-1$$

其中,  $e$  是标准偏差不变的高斯白噪声。

二维信号用二维小波分析的去噪步骤有以下 3 步:

(1) 二维信号的小波分解。选择一个小波和小波分解的层次  $N$ , 然后计算信号  $s$  到第  $N$  层的分解。





(2) 对高频系数进行阈值量化。对于从 1 到  $N$  的每一层都选择一个阈值, 并对这层的高频系数进行软阈值量化处理。

(3) 二维小波的重构。根据小波分解的第  $N$  层的低频系数和经过修改的从第一层到第  $N$  层的各层高频系数计算二维信号的小波重构。

在这 3 个步骤中, 重点是如何选取阈值和阈值的量化。

下面给出一个二维信号 (文件名为 tire.mat), 并利用小波分析对信号进行去噪处理。MATLAB 的去噪函数有 ddencomp、wdencomp 等, 其去噪过程可以按照如下程序进行。

## 2.1.4 小波分析去噪 MATLAB 例程

### 【例 2-1】小波分析去噪例程 1。

```
%装入图像
load tire
%下面进行早声的产生
init=3718025452;
rand('seed',init);
Xnoise=X+18*(rand(size(X)));
%显示原始图像及它的含噪声的图像
colormap(map);
subplot(2,2,1);image(wcodemat(X,192));
title('原始图像')
axis square
subplot(2,2,2);image(wcodemat(X,192));
title('含噪声的图像');
axis square
%用 sym5 小波对图像信号进行二层的小波分解
[c,s]=wavedec2(X,2,'sym5');
%下面进行图像的去噪处理
%使用 ddencomp 函数来计算去噪的默认阈值和熵标准
%使用 wdencomp 函数来实现图像的压缩
[thr,sorh,keepapp]=ddencomp('den','wv',Xnoise);
[Xdenoise,cxc,lxc,perf0,perf12]=wdencomp('gbl',c,s,'sym5',2,thr,sorh,keepapp);
%显示去噪后的图像
subplot(2,2,3);image(Xdenoise);
title('去噪后的图像');
axis square
```

输出结果如图 2-2 所示, 从图中 3 个图像的比较可以看出, MATLAB 中的 ddencomp 和 wdencomp 函数可以有效地进行去噪处理。

再给定一个有较大白噪声的 wmandril.mat 图像。由于图像所含的噪声主要是白噪声, 而且主要集中在图像的高频部分, 所以可以通过滤掉图像中的全部高频部分实现图像的去噪。具体去噪过程可按照如下程序进行。

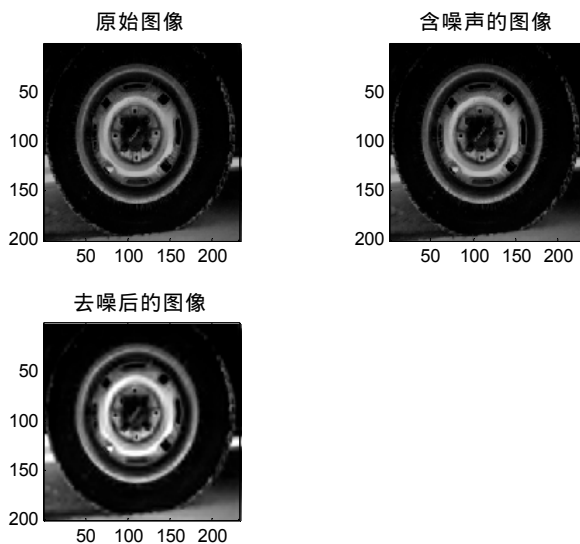


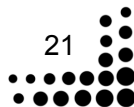
图 2-2 小波分析用于图像去噪 (一)

## 【例 2-2】小波分析去噪例程 2。

```

%下面装入原始图像，X 中含有被装载的图像
load wmandril;
%画出原始图像
subplot(221);image(X);colormap(map);
title('原始图像');
axis square
%产生含噪图像
init=2055615866;randn('seed',init)
x=X+38*randn(size(X));
%画出含噪图像
subplot(222);image(x);colormap(map);
title('含噪声图像');
axis square;
%下面进行图像的去噪处理
%用小波函数 sym4 对 x 进行 2 层小波分解
[c,s]=wavedec2(x,2,'sym4');
%提取小波分解中第一层的低频图像，即实现了低通滤波去噪
a1=wrcoef2('a',c,s,'sym4');
%画出去噪后的图像
subplot(223);image(a1);
title('第一次去噪图像');
axis square;
%提取小波分解中第二层的低频图像，即实现了低通滤波去噪
%相当于把第一层的低频图像经过再一次的低频滤波处理
a2=wrcoef2('a',c,s,'sym4',2);

```





```
%画出去噪后的图像
subplot(224);image(a2);title('第二次去噪图像');
axis square;
```

输出结果如图 2-3 所示,从图中可以看出,第一次去噪已经滤去了大部分的高频噪声,但从去噪图像与原始图像相比可以看出,第一次去噪后的图像中还是含有不少高频噪声;第二次去噪是在第一次去噪的基础上再次滤去其中的高频噪声,可以看出它具有较好的去噪效果。

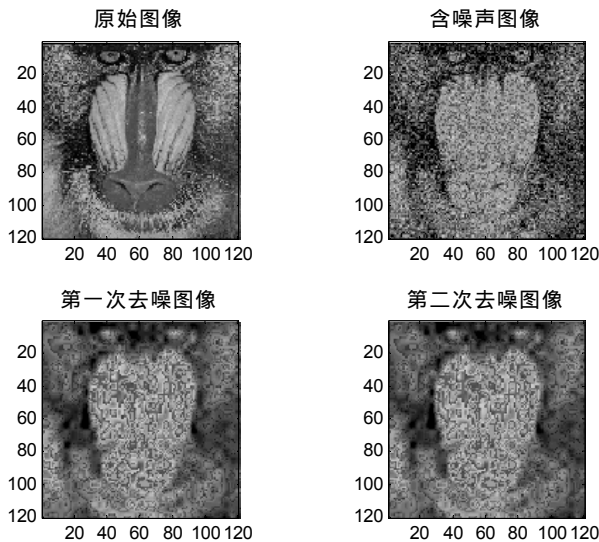


图 2-3 小波分析用于图像去噪 (二)

下面再给出定一个含有较少噪声的 `facets.mat` 图像。由于原始图像中只含有较少的高频噪声,如果按照上一个例子的方法把高频噪声全部滤掉,那么将损坏图像中固有的高频有用信号。因此这幅图像适合采用小波分解系数阈值量化方法进行去噪处理。

【例 2-3】小波分析去噪例程 3。

```
%下面装入原始图像,X 中含有被装载的图像
load facets;
%画出原始图像
subplot(221);image(X);colormap(map);
title('原始图像');
axis square
%产生含噪声图像
init=2055615866;randn('seed',init)
x=X+10*randn(size(X));
%画出含噪声图像
subplot(222);image(X);colormap(map);
title('含噪声图像');
axis square
```



```
%下面进行图像的去噪处理
%用小波函数 coif3 对 x 进行 2 层小波分解
[c,s]=wavedec2(x,2,'coif3');
%提取小波分解中第一层的低频图像，即实现了低通滤波去噪
%设置尺度向量 n
n=[1,2]
%设置阈值向量 p
p=[10.12,23.28];
%对三个方向高频系数进行阈值处理
nc=wthcoef2('h',c,s,n,p,'s');
nc=wthcoef2('v',c,s,n,p,'s');
nc=wthcoef2('d',c,s,n,p,'s');
%对新的小波分解结构[nc,s]进行重构
xx=waverec2(nc,s,'coif3');
%画出重构后图像的波形
subplot(223);image(X);colormap(map);
title('去噪后的图像');
```

axis square 输出结果如图 2-4 所示。

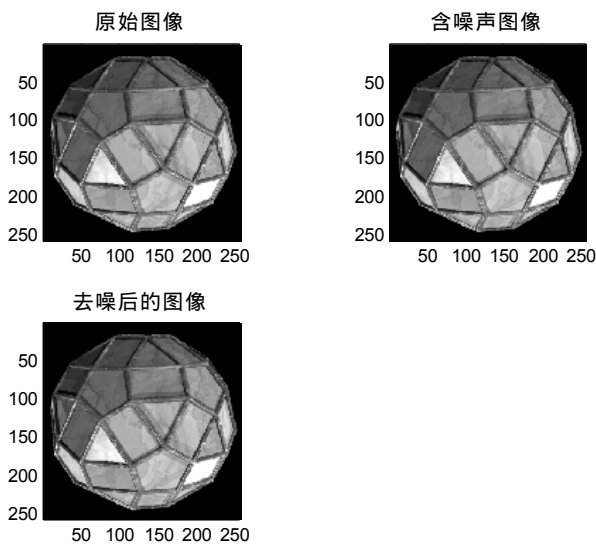
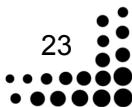


图 2-4 小波分析用于图像去噪（三）

二维信号在应用中一般表现为图像信号。二维信号在小波域中的去噪方法的基本思想与一维信号一样，在阈值选择上可以使用统一的全局阈值，又可以分作三个方向，分别是水平方向、竖直方向和对角方向，这样就可以把在所有方向的噪声分离出来，通过作用阈值抑制其成分。

**【例 2-4】**利用二维小波变换对给定图像进行小波去噪处理。

```
%装载并图示原始图像
load facets
```







```
subplot(2,2,1);
image(X);
colormap(map);
title('原始图像');

%生成含噪图像并图示
init=2055615866;
randn('seed',init);
XX=X+8*randn(size(X));
subplot(2,2,2);
image(XX);
colormap(map);
title('含噪图像');

%对图像进行去噪处理
%用小波函数 coif2 对图像 XX 进行 2 层分解
[c,l]=wavedec2(XX,2,'coif2');
%设置尺度向量
n=[1,2];
%设置阈值向量
p=[10.28,24.08];

%对高频小波系数进行阈值处理
%nc=wthcoef2('h',c,l,n,p,'s');
%nc=wthcoef2('v',c,l,n,p,'s');
nc=wthcoef2('d',c,l,n,p,'s');
%图像的二维小波重构
X1=waverec2(nc,l,'coif2');
subplot(2,2,3);
image(X1);
colormap(map);
title('第一次去噪后的图像');

%再次对高频小波系数进行阈值处理
%mc=wthcoef2('h',nc,l,n,p,'s');
mc=wthcoef2('v',nc,l,n,p,'s');
%mc=wthcoef2('d',nc,l,n,p,'s');

%%图像的二维小波重构
X2=waverec2(mc,l,'coif2');
subplot(2,2,4);
image(X2);
colormap(map);
```



```
title('第二次去噪后的图像');
```

利用不同的阈值对含噪图像去噪的结果如图 2-5 所示。第一次去噪滤去了大部分高频噪声,但与原图比较,依然有不少高频噪声;第二次去噪在第一次去噪基础上,再次滤去高频噪声,去噪效果较好,但图像质量比原图稍差。

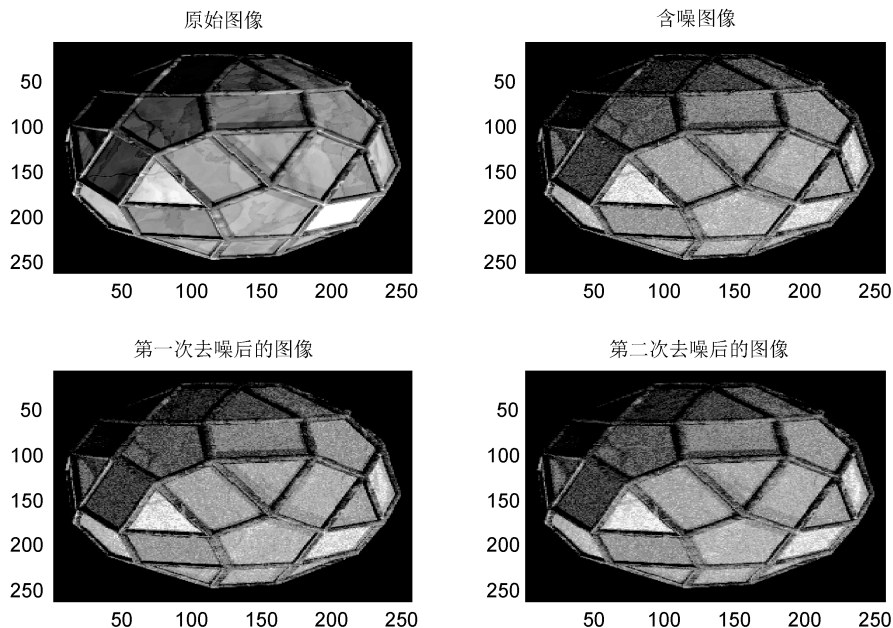


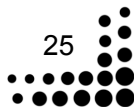
图 2-5 小波(设置不同阈值)的图像去噪结果

## 2.2 基于小波分析的图像压缩

图像压缩在图像的传输和存储中起着至关重要的作用。小波变换由于具有良好的时-频局部化性能,有效地克服了傅里叶变换在处理非平稳的复杂图像信号时所存在的局限性,因而在图像压缩领域受到了广泛的重视,已出现了许多较为成熟的算法,如 EZW 编码, SPIHT 编码等。Shapiro 利用零树处理图像小波系数,有效地利用了带间相关性和带内相关性,获得了较高的编码效率。由 Shannon 信息论理论知道,对于无记忆信源,矢量量化总是优于标量量化。给定码率时,维数任意大的矢量量化可以任意接近率失真下界。由于小波逆变换具有一定的平滑作用,小波变换域内作矢量量化不会出现明显的方块效应,所以具有较好的图像压缩效果。

### 2.2.1 图像小波分解的特点

通过水平和垂直滤波,可分离二维小波变换将原始图像分解为水平、垂直、对角和低频 4 个子带,其中低频部分可进一步分解。图像经小波变换后所得到的系数有特殊性质。在不同尺度的高频子带图像之间存在同构特性,而且 3 个方向上不同尺度下的小波系数能量大小





不同, 各方向的侧重不同。在同一方向上, 有更强的同构性和相似性。事实上, 各方向不同尺度下对应频带的相关性是最强的。为提高矢量量化的编码效率, 在构造矢量时, 必须充分利用这些相关性。此外, 图像的能量主要集中在低频子带, 高频子带所占能量较少, 且不同分辨率不同高频子带中的分布非常相似, 接近 Gamma 分布或 Laplace 分布。各高频子带系数大部分分布在零值附近, 概率密度分布曲线的中心点和最大值为零。这样, 对带内及带间相关性的充分利用和对零值附近小波系数的有效处理, 就成为提高图像压缩效率的关键。

高性能的矢量量化器必须依照图像小波系数的特性来构造矢量。使用不同子带的系数构成矢量来压缩小波系数, 就可以利用不同尺度同方向小波系数的相关性。根据以上分析, 这里采用 3 方向跨频带矢量的构造方法。小波变换将图像分解为 4 层共 13 个子带的塔形结构, 各方向以树形关系从各子图中取大小为  $4^{4-m}$  ( $m=1, 2, 3, 4$ ) 的系数块, 按图 2-6 所示的方法构成 85 维矢量。这样构造出来的跨频带矢量能够充分利用小波系数的带间和带内相关性, 但同时也带来计算量过大的问题。在图像的多分辨率分解中, 分辨率越低的频带, 其小波系数所包含的图像信息越多, 对图像重构更为重要; 而分辨率越高的频带所包含的图像信息越少。因此, 为降低矢量量化编码的复杂度, 对每个 85 维矢量的后 64 维矢量取均值, 使矢量维数大幅度减小为 22 维。这样就构造出了 3 方向小波系数的跨频带矢量。

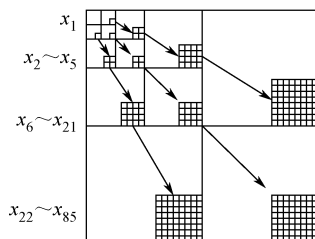


图 2-6 3 方向跨频带矢量的构造方法

### 2.2.2 小波零树和 3 方向跨频带矢量的分类

由小波图像的多分辨率解析特点, 大量的小波系数分布在零值附近, 并且具有明显的方向性, 构造出的跨频带矢量也就具有不同的能量和方向特征。通过对矢量进行分类后, 用各自独立的码书分别进行量化, 可以更有效地利用各子带间的相关性。

矢量的能量大小决定了其对于恢复图像质量的贡献程度, 矢量能量越大, 对恢复图像质量的贡献程度就越高, 所以能量可以作为矢量重要与否的一个判定准则。另外, 按四叉树规则构造矢量与零树编码的思想是一致的。零树是基于小波系数相关性的一种假设: 如果在低分辨率高频子带上的小波系数相对于阈值  $T$  是无意义或不重要的, 那么位于同方向同空间位置高分辨率子带上的小波系数相对于  $T$  在统计意义下也是无意义的。把满足这种假设的系数用树状结构表示出来就是零树。零树矢量对恢复图像质量的贡献很小。若一个零树矢量同时能量满足小于给定的能量阈值, 就可以看作非重要类, 不再进行量化编码, 把其中的每个分量都置为 0, 并用 1 个比特作标记, 记为 0; 而其他矢量均看作重要类, 记为 1, 进行较大码书尺寸的矢量量化, 以减小量化误差。采用能量阈值和零树矢量的双重判断, 既充分利用了子带相关性, 又有效地保护了图像的重要信息。



小波变换通过多分辨分析过程将一幅图像分成近似和细节部分，细节对应的是小尺度的瞬间，它在本尺度内很稳定。因此将细节存储起来，对近似部分在下一个尺度上进行分解，重复该过程即可。近似与细节在正交镜像滤波器算法中分别对应于高通和低通滤波器，这种变换通过尺度去掉相关性，在视频压缩中被证明是有效的。由于小波变换后高频部分小波系数的绝对值较小，而低频部分小波系数的绝对值较大，这样，在图像编码处理中，可以对高频部分大多数系数分配较小的比特以达到压缩的目的。

对图像进行小波分解后，可得到一系列不同分辨率的子图像（它们所对应的频率不相同）。而对于图像来说，表征它的最主要部分是低频部分，而高频部分大部分点的数值均接近于0，而且频率越高，这种现象越明显。因此，利用小波分解去掉图像的高频部分而仅仅保留图像的低频部分是一种最简单的图像压缩方法。

### 2.2.3 基于小波变换的图像局部压缩

基于离散余弦变换的图像压缩算法，其基本思想是在频域对信号进行分解，去除信号点之间的相关性，并找出重要系数，滤掉次要系数，以达到压缩的效果，但该方法在处理过程中并不能提供时域的信息，当比较关心时域特性时就无能为力了。

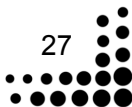
但是这种应用的需求是很广泛的，比如，遥感测控图像，要求在整幅图像有很高压缩比的同时，对热点部分的图像要有较高的分辨率；医疗图像，需要对某个局部的细节部分有很高的分辨率，单纯的频域分析的方法显然不能达到这个要求，虽然可以通过对图像进行分块分解，然后通过每块作用不同的阈值或掩码来达到这个要求，但分块大小相对固定，有失灵活。

在这方面，小波分析就优越得多。由于小波分析固有的时频特性，我们可以在时频两个方向对系数进行处理，这样就可以对我们感兴趣的部分提供不同的压缩精度。

下面我们利用小波变化的时频局部化特性，举一个局部压缩的例子，大家可以看出小波变换在这类问题上的优越性。

#### 【例 2-5】图像局部压缩。

```
load wbarb
% 使用 sym4 小波对信号进行一层小波分解
[ca1,ch1,cv1,cd1]=dwt2(X,'sym4');
codca1=wcodemat(ca1,192);
codch1=wcodemat(ch1,192);
codcv1=wcodemat(cv1,192);
codcd1=wcodemat(cd1,192);
% 将 4 个系数图像组合为 1 个图像
codx=[codca1,codch1,codcv1,codcd1]
% 复制原图像的小波系数
rca1=ca1;
rch1=ch1;
rcv1=cv1;
rcd1=cd1;
```





```
% 将 3 个细节系数的中部置零
rch1(33:97,33:97)=zeros(65,65);
rcv1(33:97,33:97)=zeros(65,65);
rcd1(33:97,33:97)=zeros(65,65);
codrca1=wcodemat(rca1,192);
codrch1=wcodemat(rch1,192);
codrev1=wcodemat(rcv1,192);
codred1=wcodemat(rcd1,192);
% 将处理后的系数图像组合为 1 个图像
codrx=[codrca1,codrch1,codrev1,codred1]
% 重建处理后的系数
rx=idwt2(rca1,rch1,rcv1,rcd1,'sym4');
subplot(221);image(wcodemat(X,192))
colormap(map);
title('原始图像');
subplot(222);image(codrx)
colormap(map);
title('一层分解后各层系数图像');
subplot(223);image(wcodemat(rx,192)),
colormap(map);
title('压缩图像');
subplot(224);image(codrx),
colormap(map);
title('处理后各层系数图像');
% 求压缩信号的能量成分
per=norm(rx)/norm(X)
per=1.0000
% 求压缩信号与原信号的标准差
err=norm(rx-X)
err =
586.4979
```

运行结果如图 2-7 所示，从图中可以看出，小波域的系数表示的是原图像各频率段的细节信息，并且提供了一种位移相关的信息表述方式，我们可以通过对局部细节系数处理来达到局部压缩的效果。

在本例中，我们把图像中部的细节系数都置零，从压缩图像中可以很明显地看出只有中间部分变得模糊（比如在原图中很清晰的围巾条纹不能分辨），而其他部分的细节信息仍然可以分辨得很清楚。

最后需要说明的是，本例只是为了演示小波分析应用在图像局部压缩的方法，在实际应用中，可能不会只做一层变换，而且作用阈值的方式可能也不会是将局部细节系数全部清除，更一般的情况是在  $N$  层变换中通过选择零系数比例或能量保留成分作用不同的阈值，实现分片的局部压缩。而且，作用的阈值可以是方向相关的，即在 3 个不同方向的细节系数上作用不同的阈值。

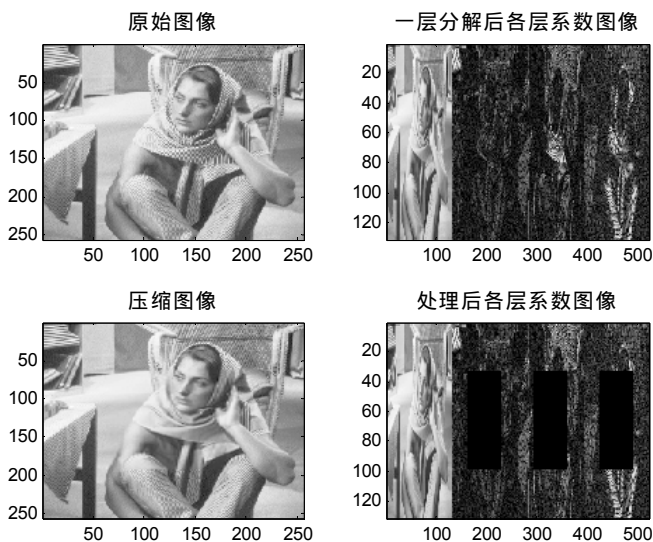


图 2-7 利用小波变换的局部压缩图像

### 2.2.4 小波变换用于图像压缩的一般方法

二维小波分析用于图像压缩是小波分析应用的一个重要方面。它的特点是压缩比高、压缩速度快，压缩后能保持图像的特征基本不变，且在传递过程中可以抗干扰。小波分析用于图像压缩具有明显的优点。

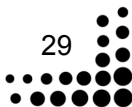
#### (1) 利用二维小波分析进行图像压缩

基于小波分析的图像压缩方法很多，比较成功的有小波包、小波变换零树压缩、小波变换矢量量化压缩等。

下面给出一个图像信号（一个二维信号，文件名为 `wbarb.mat`），利用二维小波分析对图像进行压缩。一个图像进行小波分解后，可得到一系列不同分辨率的子图像，不同分辨率的子图像对应的频率是不同的。高分辨率（高频）子图像上大部分点的数值都接近于 0，越是高频这种现象越明显。表现一个图像最主要的部分是低频部分，所以一个最简单的压缩方法是利用小波分解，去掉图像的高频部分而只保留低频部分。图像压缩可按如下程序进行处理。

#### 【例 2-6】图像压缩的一般方法。

```
%装入图像
load wbarb;
%显示图像
subplot(221);image(X);colormap(map)
title('原始图像');
axis square
disp('压缩前图像 X 的大小：');
whos('X')
%对图像用 bior3.7 小波进行 2 层小波分解
[c,s]=wavedec2(X,2,'bior3.7');
```





```

%提取小波分解结构中第一层低频系数和低频系数
ca1=appcoef2(c,s,'bior3.7',1);
ch1=detcoef2('h',c,s,1);
cv1=detcoef2('v',c,s,1);
cd1=detcoef2('d',c,s,1);
%分别对各频率成分进行重构
a1=wrcoef2('a',c,s,'bior3.7',1);
h1=wrcoef2('h',c,s,'bior3.7',1);
v1=wrcoef2('v',c,s,'bior3.7',1);
d1=wrcoef2('d',c,s,'bior3.7',1);
c1=[a1,h1,v1,d1];
%显示分解后各频率成分的信息
subplot(222);image(c1);
axis square
title('分解后低频和高频信息');
%下面进行图像压缩处理
%保留小波分解第一层低频信息，进行图像的压缩
%第一层的低频信息即为 ca1，显示第一层的低频信息
%首先对第一层信息进行量化编码
ca1=appcoef2(c,s,'bior3.7',1);
ca1=wcodemat(ca1,440,'mat',0);
%改变图像的高度
ca1=0.5*ca1;
subplot(223);image(ca1);colormap(map);
axis square
title('第一次压缩');
disp('第一次压缩图像的大小为：');
whos('ca1')
%保留小波分解第二层低频信息，进行图像的压缩，此时压缩比更大
%第二层的低频信息即为 ca2，显示第二层的低频信息
ca2=appcoef2(c,s,'bior3.7',2);
%首先对第二层信息进行量化编码
ca2=wcodemat(ca2,440,'mat',0);
%改变图像的高度
ca2=0.25*ca2;
subplot(224);image(ca2);colormap(map);
axis square
title('第二次压缩');
disp('第二次压缩图像的大小为：');
whos('ca2')

```

输出结果如下所示。

压缩前图像 X 的大小：

Name	Size	Bytes	Class
------	------	-------	-------

```

X      256x256      524288      double array
Grand total is 65536 elements using 524288 bytes
第一次压缩图像的大小为：
Name      Size      Bytes      Class
ca1      135x135      145800      double array
Grand total is 18225 elements using 145800 bytes
第二次压缩图像的大小为：
Name      Size      Bytes      Class
ca2      75x75      45000      double array
Grand total is 5625 elements using 45000 bytes

```

图像对比如图 2-8 所示，可以看出，第一次压缩提取的是原始图像中小波分解第一层的低频信息，此时压缩效果较好，压缩比较小（约为 1/3）；第二次压缩是提取第一层分解低频部分的低频部分（小波分解第二层的低频部分），其压缩比较大（约为 1/12），压缩效果在视觉上也基本过得去。这是一种最简单的压缩方法，只保留原始图像中低频信息，不经过其他处理即可获得较好的压缩效果。

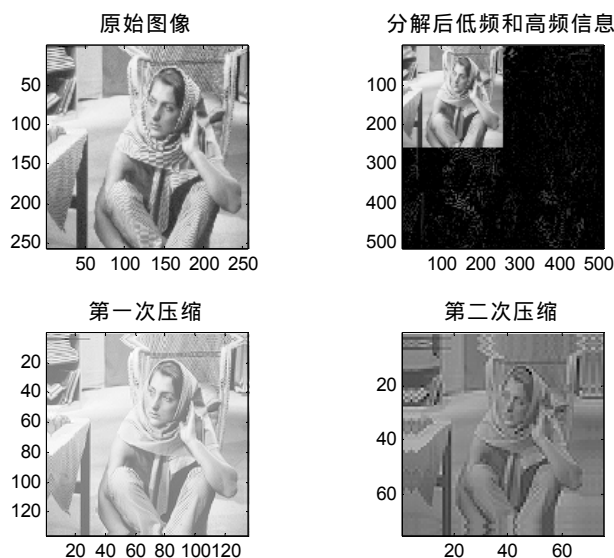


图 2-8 利用二维小波分析进行图像压缩（一）

在上面的例子中，我们还可以只提取小波分解第三、四……层的低频信息。从理论上说，我们可以获得任意压缩比的压缩图像。

下面给出一个图像信号（一个二维信号，文件名为 `tire.mat`），利用二维小波分析对图像进行压缩。

### 【例 2-7】图像压缩举例。

```

%装入一个二维信号
load tire;
%显示图像
subplot(221);image(X);colormap(map)

```





```
title('原始图像','fontsize',8);
axis square
%下面进行图像压缩
%对图像用 db3 小波进行 2 层小波分解
[c,s]=wavedec2(X,2,'db3');
%使用 wavedec2 函数来实现图像的压缩
[thr,sorh,keepapp]=ddencmp('cmp','wv',X);
%输入参数中选择了全局阈值选项 ' gbl ', 用来对所有高频系数进行相同的阈值量化处理
[Xcomp,cxc,lxc,perf0,perf12]=wdencmp('gbl',c,s,'db3',2,thr,sorh,keepapp);
%将压缩后的图像与原始图像相比较,并显示出来
subplot(222);image(Xcomp);colormap(map)
title('压缩图像','fontsize',8);
axis square
disp('小波分解系数中置 0 的系数个数百分比:');
perf0
disp('压缩后图像剩余能量百分比:');
perf12
```

程序运行结果如下所示。

```
小波分解系数中置 0 的系数个数百分比:
perf0 =48.7276
压缩后图像剩余能量百分比:
perf12 =99.9930
```

图像对比如图 2-9 所示。

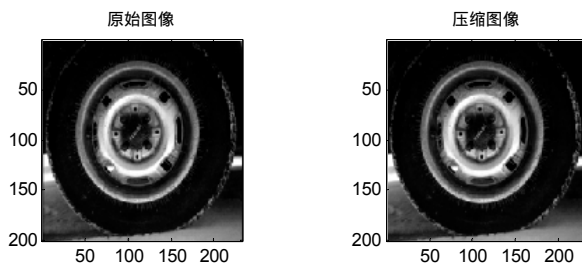


图 2-9 利用二维小波分析对图像进行压缩 (二)

利用二维小波变换进行图像压缩时,小波变换将图像从空间域变换到时间域,它的作用与以前在图像压缩中所用到的离散余弦(DCT)、傅里叶变换(FFT)等的作用类似。但是要很好地进行图像的压缩,就需要综合利用多种其他技术,特别是数据的编码与解码算法等,所以利用小波分析进行图像压缩通常需要利用小波分析和许多其他相关技术共同完成。

## (2) 二维信号压缩中的阈值的确定与作用命令

由于阈值处理只关心系数的绝对值,并不关心系数的位置,所以二维小波变换系数的阈值化方法同一维信号大同小异。为了方便用户使用小波工具箱,MATLAB 对某些阈值化方法提供了专门的二维处理命令。



下面通过一个例子来说明二维信号小波压缩的一般方法。在这个例子中，我们同时采用求默认阈值的 `ddencmp` 命令和基于经验公式的 `wdcbm2` 命令对图像进行压缩，并对压缩效果进行比较。

**【例 2-8】图像压缩举例。**

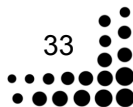
```
load detfingr;
% 求得颜色映射表的长度，以便于后面的转换
nbc=size(map,1);
% 用默认方式求出图像的全局阈值
[thr,sorh,keepapp]=ddencmp('cmp','wv',X);
thr
% 对图像作用全局阈值
[xd,cxd,lxd,perfl0,perfl2]=wdencmp('gbl',X,'bior3.5',3,thr,sorh,keepapp);
% 用 bior.3.5 小波对图像进行 3 层分解
[c,s]=wavedec2(X,3,'bior3.5');
% 指定 Birge-Massart 策略中的经验系数
alpha=1.5;m=2.7*prod(s(1,:));
% 根据各层小波系数确定分层阈值
[thr1,nkeep1]=wdcbm2(c,s,alpha,m);
% 对原图像作用分层阈值
[xd1,cxd1,sxd1,perfl01,perfl21]=wdencmp('lvd',c,s,'bior3.5',3,thr1,'s');
thr1
% 将颜色映射表转换为灰度映射表
colormap(pink(nbc));
subplot(221);
image(wcodemat(X,nbc));
title('原始图像');
subplot(222);
image(wcodemat(xd,nbc));
title('全局阈值化压缩图像');
xlabel(['能量成分',num2str(perfl2),'%','零系数成分',num2str(perfl0),'%']);
subplot(223);
image(wcodemat(xd1,nbc));
title('分层阈值化压缩图像');
xlabel(['能量成分',num2str(perfl21),'%','零系数成分', num2str(perfl01),'%']);
```

显示结果如图 2-10 所示。

可见分层阈值化压缩方法同全局阈值化方法相比，在能量损失不是很大的情况下可以获得最高的压缩化，这主要是因为层数和方向相关的阈值化方法能利用更精细的细节信息进行阈值化处理。

### (3) 基于小波包变换的图像压缩

小波分析之所以在信号处理中有着强大的功能，是基于其分离信息的思想，分离到各个小波域的信息除了与其他小波域的关联，使得处理的时候更为灵活。全局阈值化方法作用的





信息粒度太大，不够精细，所以很难同时获得高的压缩比和能量保留成分，在作用的分层阈值以后，性能明显提高，因为分层阈值更能体现信号固有的时频局部特性。

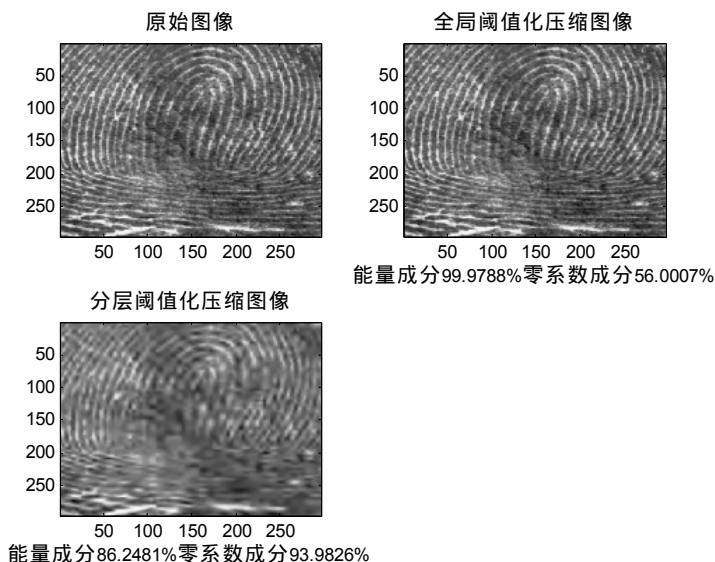


图 2-10 全局阈值化压缩和分层阈值化压缩

但是小波分解仍然不够灵活，分解出来的小波树只有一种模式，不能完全体现时-频局部化信息。而压缩的核心思想是尽可能去处各小波域系数之间的信息关联，最大程度地体现时频局部化的信息，因此，实际的压缩算法多采用小波包算法，而小波树的确定则是根据不同的信息论准则，以达到分解系数表达的信息密度最高。

下面我通过一个例子来说明小波包分析在图像压缩中的应用，并给出性能参数，以便于同基于小波分析的压缩进行比较。

#### 【例 2-9】图像压缩举例。

```
% 读入信号
load julia
% 求颜色索引表长度
nbc=size(map,1);
% 得到信号的阈值，保留层数，小波树优化标准
[thr,sorh,keepapp,crit]=ddencmp('cmp','wp',X)
% 通过以上得到的参数对信号进行压缩
[xd,treed,perf0,perf12]=wpdencmp(X,sorh,4,'sym4',crit,thr*2,keepapp);
% 更改索引表为 pink 索引表
colormap(pink(nbc));
subplot(121);
image(wcodemat(X,nbc));
title('原始图像');
subplot(122);
image(wcodemat(xd,nbc));
```



```
title('全局阈值化压缩图像');
xlabel(['能量成分',num2str(perf12),'%', '零系数成分',num2str(perf0),'%']);
plot(treed);
```

得到的压缩结果如图 2-11 所示，压缩过程中使用的最优小波树如图 2-12 所示。

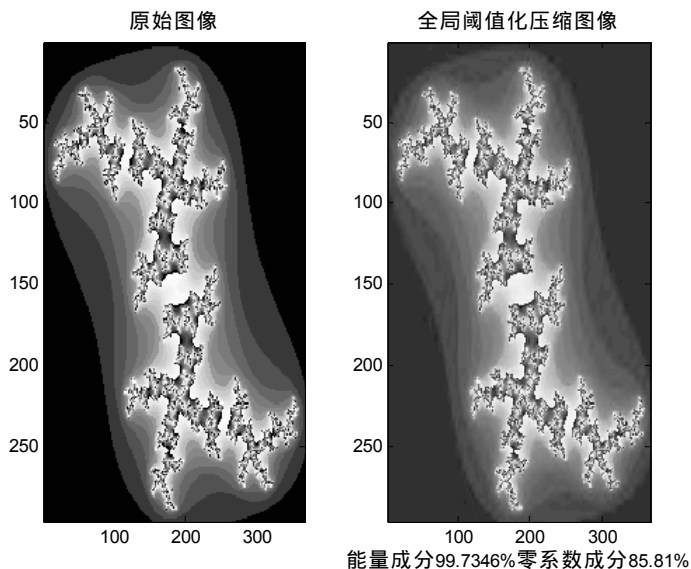


图 2-11 基于小波包分析的图像压缩

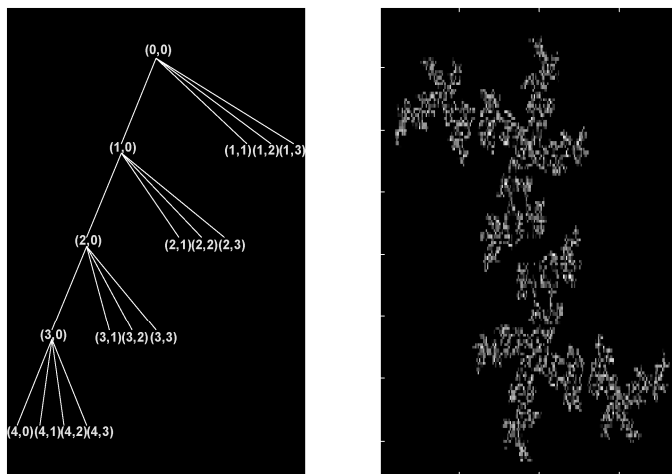
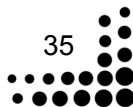


图 2-12 最优小波树

ddencmp 和 wedencmp 命令是 MATLAB 小波工具箱提供的自动获取阈值和自动使用小波包压缩的命令，后者将分解阈值化和重建综合起来。在将小波包用于信号压缩的过程中，ddencmp 命令返回的最优小波树标准都是阈值化标准。根据这个标准确定的最优小波树可以使得压缩过程的零系数成分最高，并且自动减少计算量。

最后需要说明一点，对高频成分很多的图像，小波包的分解细节信息的特点尤其能发挥





其优势。正因为这点，FBI 的指纹库采用的就是基于小波包的压缩算法 WSQ。

图像压缩是应用非常广泛的一类问题，所以其机器实现效率是至关重要的，在实际应用中，如 JPEG2000，一般不采用通常的 mallat 算法做小波分解，而是应用特定的双正交小波，利用其滤波器分布规则的特性，用移位操作来实现滤波操作。

**【例 2-10】**利用二维小波分析对给定图像进行压缩。

```
%装载并显示原始图像
load D:\MATLAB7\小波图像处理\fs08
subplot(2,2,1);
image(X);
colormap(map);
title('原始图像');
axis square;
disp('压缩前图像的大小：');
whos('X')

%对图像进行 7 层小波分解
[c,l]=wavedec2(X,2, 'bior3.7');
%提取小波分解结构中第一层的低频系数和高频系数
cA1=appcoef2(c,l, 'bior3.7',1);
%水平方向
cH1=detcoef2('h',c,l,1);
%斜线方向
cD1=detcoef2('d',c,l,1);
%垂直方向
cV1=detcoef2('v',c,l,1);

%重构第一层系数
A1=wrcoef2('a',c,l, 'bior3.7',1);
H1=wrcoef2('h',c,l, 'bior3.7',1);
D1=wrcoef2('d',c,l, 'bior3.7',1);
V1=wrcoef2('v',c,l, 'bior3.7',1);
c1=[A1 H1;V1 D1];

%显示第一层频率信息
subplot(2,2,2);
image(c1);
title('分解后的低频和高频信息');

%对图像进行压缩：保留第一层低频信息并对其进行量化编码
ca1=wcodemat(cA1,440, 'mat',0);
%改变图像高度并显示
ca1=0.1*ca1;
```



```
subplot(2,2,3);
image(ca1);
colormap(map);
title('第一次压缩后图像');
axis square;
disp('第一次压缩后图像的大小 : ');
whos('ca1')
%压缩图像：保留第二层低频信息并对其进行量化编码
cA2=appcoef2(c,l,'bior3.7',2);
ca2=wcodemat(cA2,440,'mat',0);
ca2=0.1*ca2;
subplot(2,2,4);
image(ca2);
colormap(map);
title('第二次压缩后图像');
disp('第二次压缩后图像大小 : ');
whos('ca2')
```

压缩后的图像如图 2-13 所示。

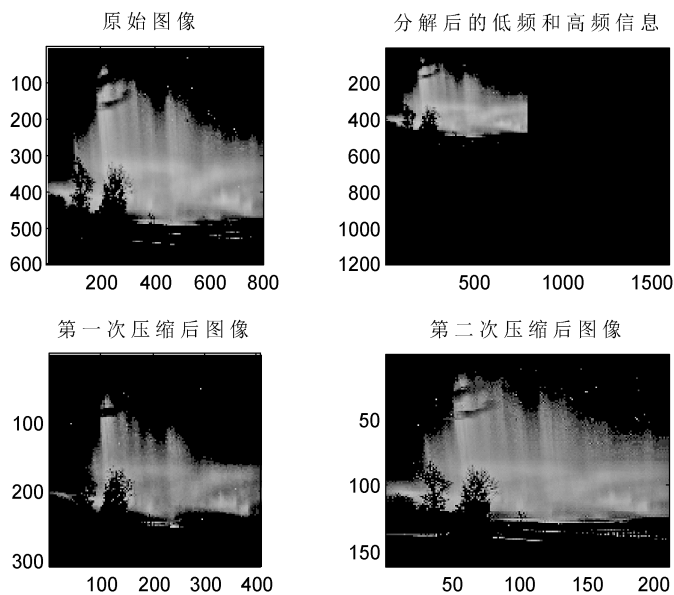


图 2-13 基于小波的图像压缩结果

输出结果如下所示：

压缩前图像的大小：

Name	Size	Bytes	Class
X	600x800	3840000	double array

Grand total is 480000 elements using 3840000 bytes

第一次压缩后图像的大小：



Name	Size	Bytes	Class
ca1	307x407	999592	double array

Grand total is 124949 elements using 999592 bytes

第二次压缩后图像大小：

Name	Size	Bytes	Class
ca2	161x211	271768	double array

Grand total is 33971 elements using 271768 bytes

从输出结果来看，第一次压缩提取原始图像中小波分解第一层的低频信息，此时压缩效果较好，压缩比较小，约为  $1/4$ ；第二次压缩是提取第一层分解低频部分的低频部分，即第二低频部分，其压缩比较大，约为  $1/14$ ，压缩效果在视觉上也基本过得去。随着分解层数的增加，压缩比是递减的。

保留原始图像中低频信息的压缩方法只是一种最简单的压缩方法。它不需要经过其他处理即可获得较好的压缩效果。当然，对于上面的例子我们还可以提取小波分解的更高层低频信息。从理论上说，可以获得任意压缩比的压缩图像，只不过在对压缩比和图像质量都有较高要求时，它就不如其他编码方法了。

## 第 3 章 小波包算法分析与应用

### 3.1 小波包与信号去噪

#### 3.1.1 基本原理

在小波包分析中，其信号去噪的算法思想和小波分析中的基本相同，所不同的就是小波包提供了一种更为复杂也更为灵活的分析手段。因为小波包分析对上层的低频部分和高频部分同时进行分解，具有更加精确的局部分析能力。

对信号进行小波包分解时，可以采用多种小波包基。通常根据分析信号的要求，从中选择最好的一种小波包基，即最优基。最优基的选择标准是熵标准。在 MATLAB 的小波工具箱中可通过 `besttree` 函数进行最优基的选择，即计算最优树。

应用小波包分析对信号进行去噪处理是它的一个最基本的功能。一般地，按照如下步骤进行：

(1) 信号的小波包分解。选择一个小波并确定所需分解的层次，然后对信号进行小波包分解。

(2) 确定最优小波包基。对于一个给定的熵标准，计算最优树（这一步不是必需的，可根据不同的目的进行有选择性的使用）。

(3) 小波包分解系数的阈值量化。对于每个小波包分解系数，都选择一个恰当的阈值并对系数进行阈值量化。

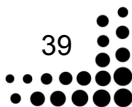
(4) 信号的小波包重构。根据最低层的小波包分解系数和经过量化处理系数，进行小波包重构。

在上述的各步中，最关键的是如何选取阈值和如何进行阈值量化，在一定程序上，它直接关系到对信号进行去噪处理的质量。

图像的多分辨率分析是一种在不同的分辨率下处理图像中不同信息的方法，它将图像在各种分辨率下的细节提取出来，得到一个拥有不同分辨率的图像细节序列后再进行图像的各种处理。

在本小节中，小波包分析进行图像去噪处理的基本原理和方法与前面所介绍的相同，仅以具体的实例来说明小波包分析在图像去噪处理中的应用。

在的小波工具箱中，提供了一个函数 `wpdencmp`，它是专门利用小波包分解实现去噪和压缩处理的。







语法格式如下：

```
[XD,TREED,PERFO,PERFL2]=wpdencmp(X,SORH,N,'wname',CRIT,PAR,KEEPAPP)
[XD,TREED,PERFO,PERFL2]=wpdencmp(TREE,SORH,CRIT,PAR,KEEPAPP)
```

使用说明：利用小波包实现信号和图像的去噪或压缩。

输入参数：sorrh 指定选取软阈值（sorrh='s'）或硬阈值（sorrh='h'）；N 为小波分解的层数；wavename 指定分解时所用的小波；CRIT 和 PAR 定义了熵准则；TREE 是小波包分解树结构。

输出参数：perfo、perfl2 返回压缩比例系数。

### 3.1.2 MATLAB 例程分析

【例 3-1】利用小波包分析对一个给定的含噪信号进行去噪处理。

```
%装载原始信号并图示之
load noismima;
s=noismima(1:1000);
figure(1);
subplot(2,2,1);
plot(s);
xlabel('样本序号 n');
ylabel('幅值 A');
title('原始信号');

%采用默认阈值、用 wdencomp 函数进行去噪处理
[thr,sorrh,keepapp,crit]=ddencmp('den','wp',s);

%用全局阈值选项进行去噪处理
[c,treed,perf0,perf12]= wpdencmp(s,sorrh,3,'db2',crit,thr,keepapp);
subplot(2,2,3);
plot(c);
xlabel('样本序号 n');
ylabel('幅值 A');
title('默认阈值去噪信号');

%根据前面的去噪效果，调节阈值大小进行去噪
thr=thr+15;
[c1,treed,perf0,perf12]= wpdencmp(s,sorrh,3,'db2',crit,thr,keepapp);
subplot(2,2,4);
plot(c1);
xlabel('样本序号 n');
ylabel('幅值 A');
title('调节阈值后的去噪信号');
```

程序运行结果如图 3-1 所示，可以明显看出，小波包很好地消除了原始信号中的噪声。

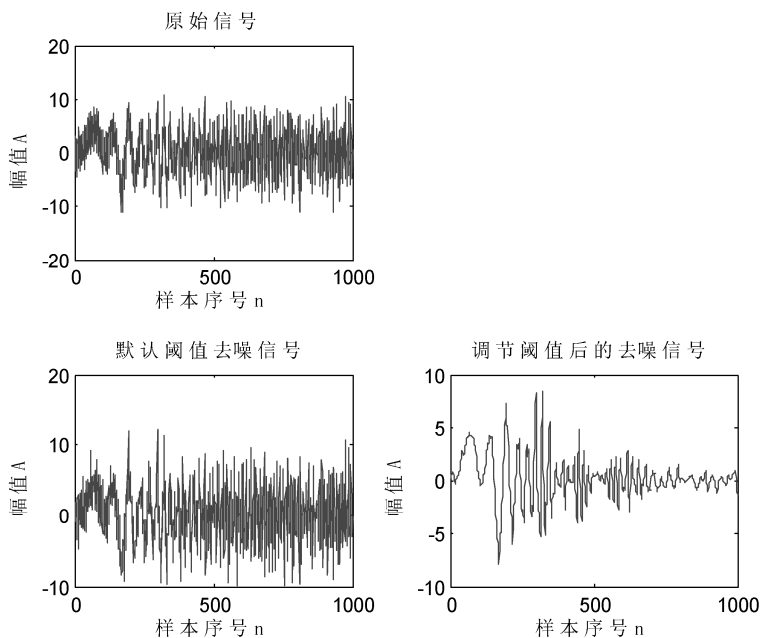


图 3-1 小波包用于信号去噪

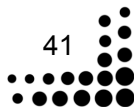
【例 3-2】利用小波包变换对一个二维含噪图像进行去噪处理。

```
%装载并显示原始图像
load julia;
subplot(2,2,1);
image(X);
colormap(map);
title('原始图像');
axis square;

%在图像中加入噪声
init=2055615866;
randn('seed',init);
X1=X+10*randn(size(X));
subplot(2,2,2);
image(X1);
colormap(map);
title('含噪图像');
axis square;

%基于小波包的去噪处理
thr=10; sorh='s';
crit='shannon';
keepapp=0;
X2=wpdencmp(X1,sorh,3,'sym4',crit,thr,keepapp);

%画出去噪后的图像
```





```
subplot(2,2,3);
image(X2);
colormap(map);
title('全局阈值去噪图像');
axis square;

%对图像进行平滑处理以增强去噪效果（中值滤波）
for i=2:175;
    for j=2:259
        Xtemp=0;
        for m=1:3
            for n=1:3
                Xtemp=Xtemp+X2(i+m-2,j+n-2);
            end
        end
        Xtemp=Xtemp/9;
        X3(i,j)=Xtemp;
    end
end

%显示平滑结果
subplot(2,2,4);
image(X3);
colormap(map);
title('平滑后的图像');
axis square;
```

程序运行结果如图 3-2 所示，其中左上角是原始图像，右上角是添加噪声后的图像，

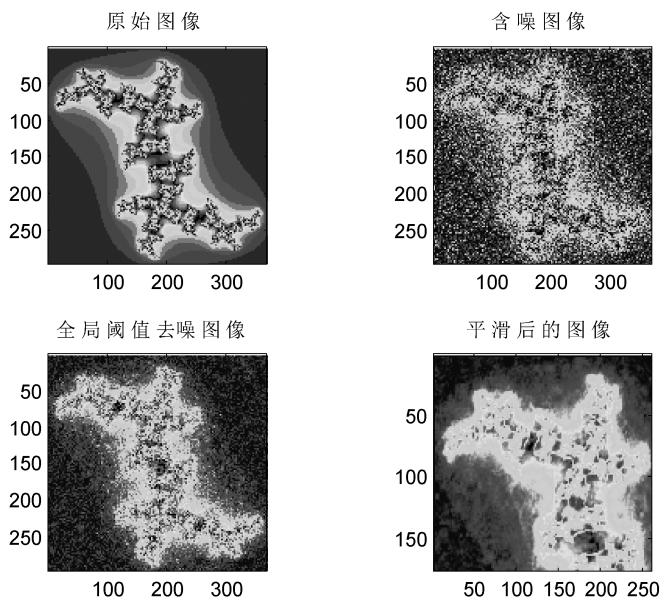


图 3-2 基于小波包变换的图像去噪



通过小波包分解并设置全局阈值去噪后的结果如左下图所示,与含噪图像相比,明显清楚了很多;进一步对去噪后的图像进行平滑处理,如右下图所示,与去噪后的图像相比,明显光滑了。

除了利用函数 `wpdencmp` 对图像进行去噪外,还可以利用二维小波包分解函数 `wpdec2` 来实现图像去噪。

**【例 3-3】**利用二维小波包分解对一个二维含噪图像进行去噪处理。

```
%装载并显示原始图像
load trees;
subplot(2,2,1);
image(X);
colormap(map);
title('原始图像');
axis square;

%生成含噪图像
init=2055615866;
randn('seed',init);
X1=X+20*randn(size(X));
subplot(2,2,2);
image(X1);
colormap(map);
title('含噪图像');
axis square;

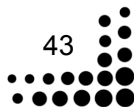
%用小波 sym2 对图像 X1 进行 1 层小波包分解
T=wpdec2(X1,1,'sym2');

%设置阈值
thr=8.342;

%对图像的小波包分解系数进行软阈值量化
NT=wpthcoef(T,0,'s',thr);

%仅对低频系数进行重构
X2=wprcoef(NT,1);

%画出去噪后的图像
subplot(2,2,3);
image(X2);
colormap(map);
title('去噪后的图像');
axis square;
```





程序运行结果如图 3-3 所示，其中左上角是原始图像，右上角是添加噪声后的图像，通过二维小波包分解后并对系数进行阈值化处理，重构的图像如左下图所示，与含噪图像相比，明显清楚了很多，达到了去噪的效果。

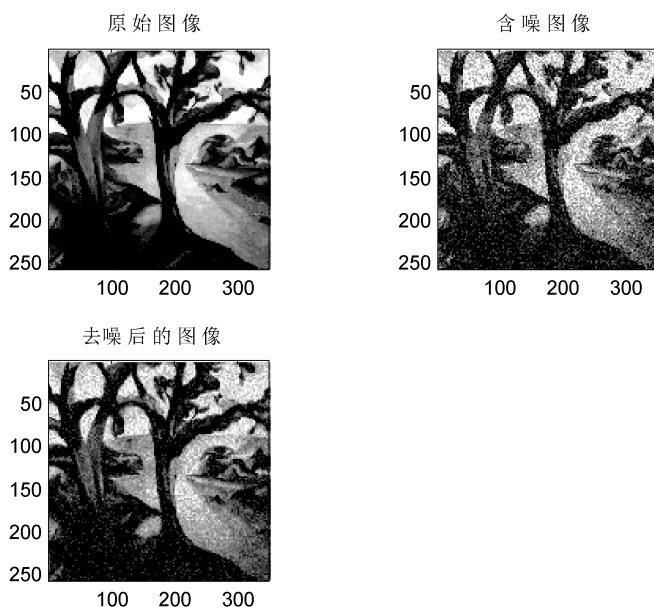


图 3-3 基于二维小波包分解的图像去噪

## 3.2 小波包分析用于信号压缩

### 3.2.1 基本原理

在小波包分析中，其信号压缩的算法思想和在小波分析中的基本相同，所不同的就是小波包提供了一种更为复杂也更为灵活的分析手段。因为小波包分析对上层的低频部分和高频部分同时进行分解，具有更加精确的局部分析能力。

在用小波包分析进行信号压缩处理中，最关键的是如何进行阈值量化，在一定的程序上，这直接关系到对信号进行压缩处理的质量。

由 Kunt 等人提出的第二代图像数据压缩算法，充分考虑了人类视觉生理、心理特征，侧重于将原始图像在频率内做多层分解，然后对这些信息表灵活地有选择地编码，可得到高压缩比和很小的失真度，比第一代图像数据压缩算法仅考虑图像本身的效果好。小波包具有能将空间精细分解的性质，所以很适合于进行图像数据的压缩。

在本节中，小包波分析进行图像压缩处理的基本原理和方法与前面所介绍的对信号压缩的相同，仅以具体的实例来说明小波包在图像压缩处理中的应用。

## 3.2.2 MATLAB 例程分析

【例 3-4】利用小波包分析对一个给定的一维信号进行压缩处理。

```
%装载源信号
load noisbump;
s=noisbump(1:1000);
figure(1);
subplot(2,1,1);
plot(s);
xlabel('样本序号 n');
ylabel('幅值 A');
title('原始信号');
%采用默认阈值，以小波包函数 wpdencmp 对 s 进行压缩处理
[thr,sorh,keepapp,crit]=ddencmp('cmp','wp',s);
[sc,treed,perf0,perf12]= wpdencmp(s,sorh,3,'db2',crit,thr,keepapp);
subplot(2,1,2);
plot(sc);
xlabel('样本序号 n');
ylabel('幅值 A');
title('压缩后的信号');
```

程序运行结果如图 3-4 所示。

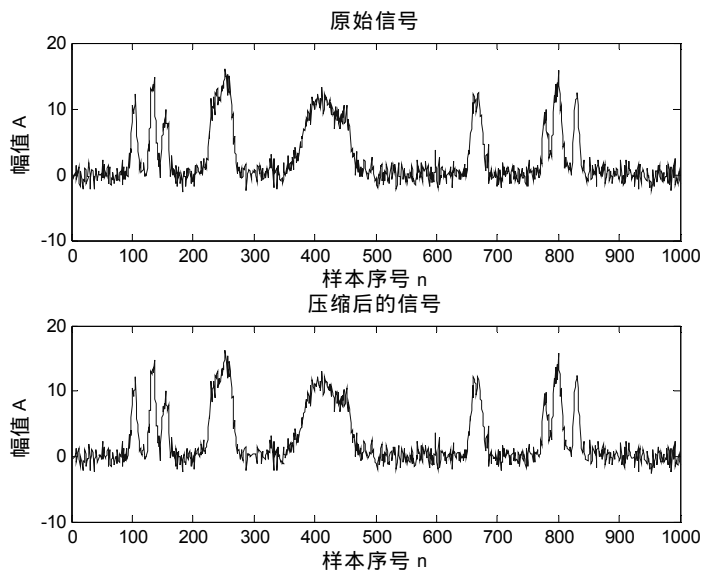


图 3-4 小波包用于压缩

```
>> perf0
perf0 =
```



```
43.5084
>> perf12
perf12 =
99.7278
```

这说明上述压缩过程保留了 99.7278% 的能量，压缩率为 43.5084%。

【例 3-5】利用小波包分析对给定图像进行压缩处理。

```
%装载并显示原始图像
load wbarb;
subplot(1,2,1);
image(X);
colormap(map);
title('原始图像');

%采用默认的全局阈值
[thr,sorh,keepapp,crit]=ddencmp('cmp','wp',X);
%图像进行压缩
Xc=wpdencmp(X,sorh,3,'bior3.1',crit,thr,keepapp);

%显示压缩结果
subplot(1,2,2);
image(Xc);
colormap(map);
title('全局阈值压缩图像');
```

程序运行结果如图 3-5 所示，其中左图是原始图像，右图是经过全局阈值压缩的图像，比较可见，压缩后的图像基本上保持了原来图像的内容，但其中包含的大量冗余信息被剔除了。

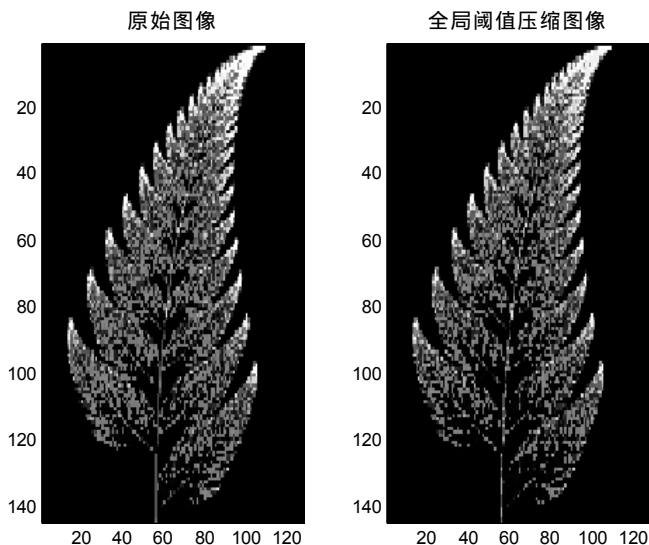


图 3-5 基于小波包分解的图像压缩（一）

### 【例 3-6】利用小波包进行图像压缩处理。

```
%装载并显示原始图像
load clown;
subplot(1,2,1);
image(X);
colormap(map);
title('原始图像');
%采用默认的全局阈值
[thr,sorh,keepapp,crit]=ddencmp('cmp','wp',X);
%图像压缩
[Xc,treed,perf0,perf12]=wpdencmp(X,sorh,3,'bior3.1',crit,thr,keepapp);
subplot(1,2,2);
image(Xc);
colormap(map);
title('压缩后的图像');
%给出压缩效率
disp('小波分解系数中置 0 的系数个数百分比 : ');
perf0
disp('压缩后图像剩余能量百分比 : ');
perf12
```

程序运行结果如图 3-6 所示，其中左图是原始图像，右图是经过全局阈值压缩过的图像。

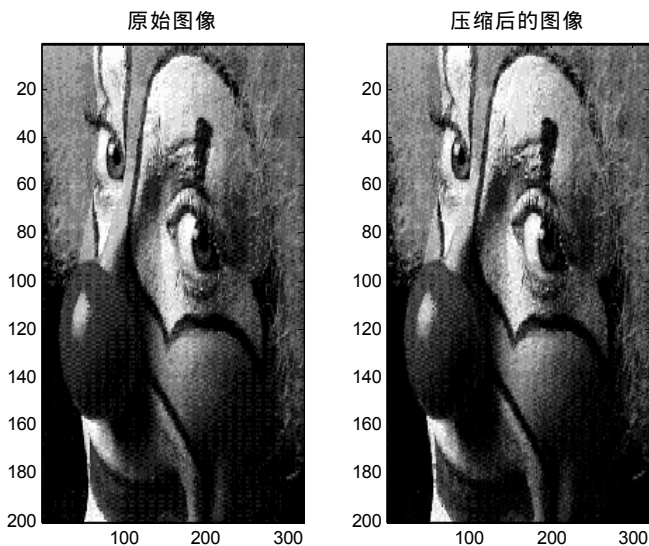


图 3-6 基于小波包分解的图像压缩（二）

```
小波分解系数中置 0 的系数个数百分比 :
perf0 =
    51.2385
压缩后图像剩余能量百分比 :
perf12 =
    99.9685
```





## 3.3 小波包与图像边缘检测

### 3.3.1 基本原理

图像的边缘检测是对图像进行进一步处理和识别的基础,虽然图像边缘产生的原因不同,但反映在图像的组成基元上,它们都是图像上灰度的不连续点或灰度剧烈变化的地方,这就意味着图像边缘就是信号的高频部分。因此所有的边缘检测方法都是检测信号的高频分量,但是在实际图像中,由于噪声的存在,边缘检测成为一个难题。

小波包分解后得到的图像序列由近似部分和细节组成,近似部分是原始图像对高频部分进行滤波所得的近似表示。经滤波后,近似部分去除了高频分量,因此能够检测到原始图像中所检测不到的边缘。

### 3.3.2 MATLAB 例程分析

【例 3-7】利用小波包进行图像边缘检测。

```
%装载并显示原始图像
load vcosig;
%加入含噪
init=2055615866;
randn('seed',init);
X1=X+20*randn(size(X));
figure(1);
image(X1);
colormap(map);
title('原始图像');
axis square;
%用小波 db4 对图像 X 进行一层小波包分解
T=wpdec2(X1,1,'db4');
%重构图像近似部分
figure(2);
A=wprcoef(T,[1 0]);
image(A);
title('图像的近似部分');
axis square;
%边缘检测
figure(3);
%%原图像的边缘检测
BW1 = edge(X1,'prewitt');
imshow(BW1);
```



```
title('原图像的边缘');
axis square;
%%图像近似部分的边缘检测
figure(4);
BW2= edge(A,'prewitt');
imshow(BW2);
title('图像近似部分的边缘');
axis square;
```

原始含噪图像如图 3-7 所示，利用 db4 正交小波基对其进行一层小波包分解后，重构其近似部分如图 3-8 所示，比较可见，经小波包分解后所得到的近似部分比原始图像层次更加分明，因此利用分解后的近似图像能检测边缘。如图 3-9 所示是直接对原始图像进行边缘检测的结果，如图 3-10 所示是对近似图像进行边缘检测的结果，比较可见，后一种方法的效果更好。

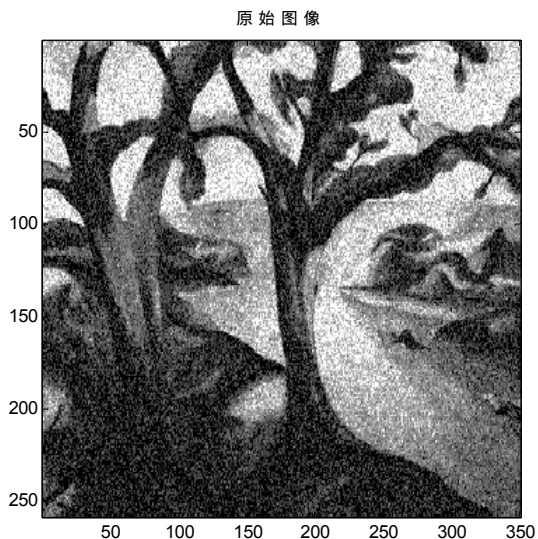


图 3-7 原始含噪图像

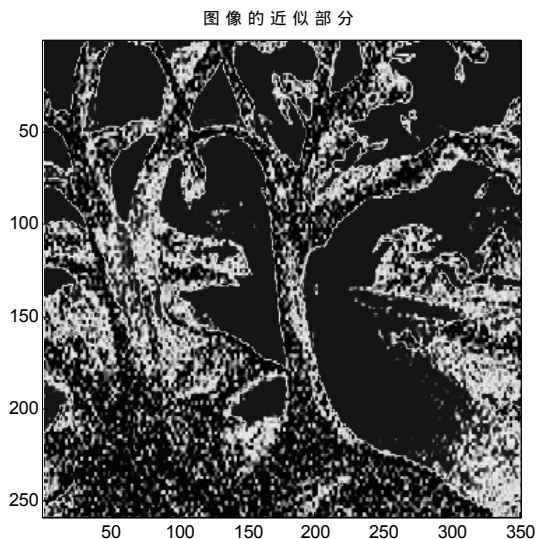


图 3-8 图像小波包分解后的近似部分



图 3-9 原始图像的边缘检测

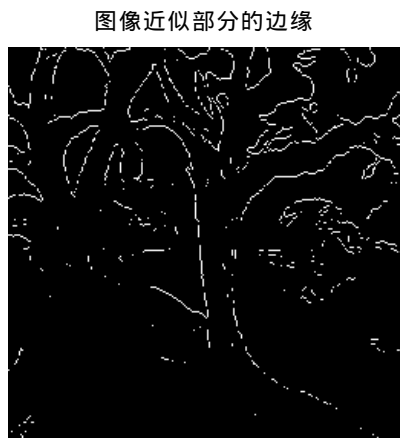
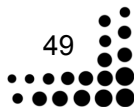


图 3-10 近似图像的边缘检测



## 第 4 章 小波快速算法设计 原理与实现

小波分析是近 20 年迅猛发展起来的一门新兴的交叉性学科，已被广泛应用于数值分析、信号处理、图像处理、量子理论、地震勘探、语音识别、计算机视觉、CT 成像、机械故障诊断等领域，并深深地影响着这些领域。它具有重要的理论价值和实际应用价值，是众多学科所共同关注的热点之一。

### 4.1 绪论

#### 4.1.1 概述

小波分析是近 20 年发展起来的一种新的时频分析方法，其典型应用包括齿轮变速控制、起重机的非正常噪声分析、自动目标锁定、物理中的间断现象分析等。而频域分析的着眼点在于区分突发信号和稳定信号以及定量分析其能量，典型应用包括细胞膜识别、金属表面探伤、金融学中快变量的检测、Internet 流量控制等。

从以上信号分析的典型应用可以看出，时频分析应用非常广泛，涵盖了物理学、工程技术、生物科学、经济学等众多领域。而且，在很多情况下单单分析其时域或频域的性质是不够的，比如在电力监测系统中，既要监控稳定信号的成分，又要准确定位故障信号。这就需要引入新的时频分析方法，小波分析正是由于这类需求发展起来的。

在传统的傅里叶分析中，信号完全是在频域展开的，不包含任何时频信息，这对于某些应用来说是很恰当的，因为信号的频率的信息对其是非常重要的。但其丢弃的时域信息可能对某些应用同样非常重要，所以人们对傅里叶分析进行了推广，提出了很多能表征时域和频域信息的信号分析方法，如短时傅里叶变换、Gabor 变换、时频分析、小波变换等。其中，短时傅里叶变换是在傅里叶分析基础上引入时域信息的最初尝试，其基本假定是在一定的时间窗内信号是平稳的，那么通过分割时间窗，在每个时间窗内都把信号展开到频域就可以获得局部的频域信息，但是它的时域区分度只能依赖于大小不变的时间窗，对某些瞬态信号来说还是粒度太大。换言之，短时傅里叶分析只能在一个分辨率上进行，所以对很多应用来说不够精确，存在很大的缺陷。

而小波分析则克服了短时傅里叶变换在单分辨率上的缺陷，具有多分辨率分析的特点，在时域和频域都有表征信号局部信息的能力，时间窗和频率窗都可以根据信号的具体形态动



态调整。在一般情况下，在低频部分（信号较平稳）可以采用较低的时间分辨率，而提高频率的分辨率，在高频情况下（频率变化不大）可以用较低的频率分辨率来换取精确的时间定位。因为这些特定，小波分析可以探测正常信号中的瞬态，并展示其频率成分，被称为数学显微镜，广泛应用于各个时频分析领域。

小波分析在图像处理中有非常重要的应用，包括图像压缩、图像去噪、图像融合、图像分解、图像增强等。本节给出了详细的程序范例，用 MATLAB 实现了基于小波变换的各种处理技术。

### 4.1.2 傅里叶变换与小波变换的比较

小波分析是傅里叶分析思想方法的发展与延拓。它自产生以来，就一直与傅里叶分析密切相关。它的存在性证明，小波基的构造以及结果分析都依赖于傅里叶分析，二者是相辅相成的。两者相比较主要有以下不同：

(1) 傅里叶变换的实质是把能量有限信号  $f(t)$  分解到以  $\{e^{j\omega t}\}$  为正交基的空间上；小波变换的实质是把能量有限信号  $f(t)$  分解到  $W_{-j}$  ( $j=1, 2, \dots, J$ ) 和  $V_{-j}$  所构成的空间上去。

(2) 傅里叶变换用到基本函数只有  $\sin \omega t, \cos \omega t, \exp i\omega t$ ，具有唯一性；小波分析用到的函数（小波函数）则具有不唯一性，同一个工程问题用不同的小波函数进行分析有时结果相差甚远。小波函数的选用是小波分析应用到实际中的一个难点问题（也是小波分析研究的一个热点问题），目前往往是通过经验或不断地试验（对结果进行对照分析）来选择小波函数。

(3) 在频域中，傅里叶变换具有较好的局部化能力，特别是对于那些频率成分比较简单的确定性信号，傅里叶变换很容易把信号表示成各频率成分的叠加和的形式，如  $\sin \omega_1 t + 0.345 \sin \omega_2 t + 4.23 \cos \omega_3 t$ 。但在时域中，傅里叶变换没有局部化能力，即无法从信号  $f(t)$  的傅里叶变换  $F(\omega)$  中看出  $f(t)$  在任一时间点附近的性态。事实上， $F(\omega)d\omega$  是关于频率为  $\omega$  的谐波分量的振幅，在傅里叶展开式中，它是由  $f(t)$  的整体性态所决定的。

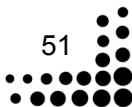
(4) 在小波分析中，尺度  $a$  的值越大，相当于傅里叶变换中  $\omega$  的值越小。

(5) 在短时傅里叶变换中，变换系数  $S(\omega, \tau)$  主要依赖于信号在  $[\tau - \delta, \tau + \delta]$  片段中的情况，时间宽度是  $2\delta$ （因为  $\delta$  是由窗函数  $g(t)$  唯一确定的，所以  $2\delta$  是一个定值）。在小波变换中，变换系数  $W_f(a, b)$  主要依赖于信号在  $[b - a\Delta\psi, b + a\Delta\psi]$  片段中的情况，时间宽度是  $2a\Delta\psi$ ，该时间宽度是随着尺度  $a$  变化而变化的，所以小波变换具有时间局部分析能力。

(6) 若用信号通过滤波器来解释，小波变换与短时傅里叶变换不同之处在于：对短时傅里叶变换来说，带通滤波器的带宽  $\Delta f$  与中心频率  $f$  无关；相反，小波变换带通滤波器的带宽  $\Delta f$  则正比于中心频率  $f$ ，即

$$Q = \frac{\Delta f}{f} = C \quad C \text{ 为常数}$$

亦即滤波器有一个恒定的相对带宽，称之为等  $Q$  结构（ $Q$  为滤波器的品质因数，且有  $Q = \text{中心频率} / \text{带宽}$ ）。





### 4.1.3 小波分析与多分辨分析的历史

小波理论包括连续小波和二进小波变换,在映射到计算域时存在很多问题,因为两者都存在信息冗余,在对信号采样以后,需要计算的信息量还是相当的大,尤其是连续小波变换,因为要对精度内所有的尺度和位移都做计算。而二进小波变换虽然在离散的尺度上进行伸缩和平移,但是小波之间没有正交性,各个分量的信息掺杂在一起,为我们的分析带来不便。

真正使小波在应用领域得到比较大发展的是 Meyer 在 1986 年提出的一组小波,其二进制伸缩和平移构成  $L^2(R)$  的标准化正交基。在此结果基础上,1988 年 S.Mallat 在构造正交小波时提出了多分辨分析的概念,从函数分析的角度给出了正交小波的数学解释,在空间的概念上形象的说明了小波的多分辨率特性,给出了通用的构造正交小波的方法,并将之前所有的正交小波构造方法统一起来,并类似于傅里叶分析中的快速傅里叶算法,给出了小波变换的快速算法——Mallat 算法。这样,在计算上变得可行以后,小波变换在各个领域才发挥它独特的优势,解决了各类问题,为人们提供了更多关于时域分析的信息。

形象一点说,多分辨分析就是要构造一组函数空间,每组空间的构成都有一个统一的形式,而所有空间的闭包则逼近  $L^2(R)$ 。在每个空间中,所有的函数都构成该空间的标准化正交基,而所有函数空间的闭包中的函数则构成  $L^2(R)$  的标准化正交基,那么,如果对信号在这类空间上进行分解,就可以得到相互正交的时频特性。而且由于空间数目是无限可数的,可以很方便地分析我们所关心的信号的某些特性。

下面简要介绍多分辨分析的数学理论。

定义 1 空间  $L^2(R)$  中的多分辨分析是指  $L^2(R)$  满足如下性质的一个空间序列  $\{V_j\}_{j \in \mathbb{Z}}$  :

- (1) 调一致性:  $V_j \subset V_{j+1}$ , 对任意  $j \in \mathbb{Z}$  ;
- (2) 渐进完全性:  $\bigcup_{j \in \mathbb{Z}} V_j = \Phi$ ,  $\text{close} \left\{ \bigcup_{j \in \mathbb{Z}} V_j \right\} = L^2(R)$  ;
- (3) 伸缩完全性:  $f(t) \in V_j \Leftrightarrow f(2t) \in V_{j+1}$  ;
- (4) 平移不变性:  $\forall k \in \mathbb{Z}, \phi(2^{-j/2}t) \in V_j \Rightarrow \phi_j(2^{-j/2}t - k) \in V_j$  ;
- (5) Riesz 基存在性: 存在  $\phi(t) \in V_0$ , 使得  $\{\phi_j(2^{-j/2}t - k) | k \in \mathbb{Z}\}$  构成  $V_j$  的 Riesz 基。关于

Riesz 的具体说明如下:

若  $\phi(t)$  是  $V_0$  的 Riesz 基, 则存在常数  $A$ 、 $B$ , 且使得

$$A \|\{c_k\}\|_2^2 \leq \left\| \sum c_k \phi(t - k) \right\|_2^2 \leq B \|\{c_k\}\|_2^2$$

对所有双无限可平方和序列  $\{c_k\}$ , 即

$$\|\{c_k\}\|_2^2 = \sum_{k \in \mathbb{Z}} |c_k|^2 < \infty$$

成立。

满足上述个条件的函数空间集合成为一个多分辨分析, 如果  $\phi(t)$  生成一个多分辨分析, 那么称  $\phi(t)$  为一个尺度函数。



可以用数学方法证明,若 $\phi(t)$ 是 $V_0$ 的Riesz基,那么存在一种方法可以把 $\phi(t)$ 转化为 $V_0$ 的标准化正交基。这样,我们只要能找到构成多分辨分析的尺度函数,就可以构造出一组正交小波。多分辨分析构造了一组函数空间,这组空间是相互嵌套的,即

$$L \subset V_{-2} \subset V_{-1} \subset V_0 \subset V_1 \subset V_2$$

那么,相邻的两个函数空间的差就定义了一个由小波函数构成的空间,即

$$V_j \oplus W_j = V_{j+1}$$

为了说明这些性质,首先来介绍双尺度差分方程,由于对 $\forall j$ ,有 $V_j \subset V_{j+1}$ ,所以对 $\forall g(x) \in V_j$ ,都有 $g(x) \in V_{j+1}$ ,也就是说,可以展开成 $V_{j+1}$ 上的标准化正交基。由于 $\phi(t) \in V_0$ ,所以 $\phi(t)$ 就可以展开成

$$\phi(t) = \sum_{n \in \mathbb{Z}} h_n \phi_{1,n}(t)$$

这就是著名的双尺度差分方程,双尺度差分方程奠定了正交小波变换的理论基础,从数学上可证明,对于任何尺度的 $\phi_{j,0}(t)$ ,它在 $j+1$ 尺度正交基 $\phi_{j+1,n}(t)$ 上的展开系数 $h_n$ 是一定的,这就为我们提供了一个很好的构造多分辨分析的方法。在频域中,双尺度差分方程的表现形式为

$$\hat{\phi}(2\omega) = H(\omega)\hat{\phi}(\omega)$$

如果 $\hat{\phi}(\omega)$ 在 $\omega=0$ 连续,则有

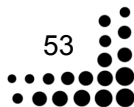
$$\hat{\phi}(\omega) = \sum_{j=1} H\left(\frac{\omega}{2^j}\right)\hat{\phi}(0)$$

说明 $\hat{\phi}(\omega)$ 的性质完全由 $\hat{\phi}(0)$ 决定。

## 4.2 从傅里叶变换到小波变换

小波分析属于时频分析的一种,传统的信号分析是建立在傅里叶变换基础上的,由于傅里叶分析使用的是一种全局的变换,要么完全在时域,要么完全在频域,因此无法表述信号的时频局域性质,而这种性质恰恰是非平稳信号最根本和最关键的性质。为了分析和处理非平稳信号,人们对傅里叶分析进行了推广乃至根本性的革命,提出并发展了一系列新的信号分析理论:短时傅里叶变换、Gabor变换、时频分析、小波变换、分数阶傅里叶变换、线调频小波变换、循环统计量理论和调幅-调频信号分析等。其中,短时傅里叶变换和小波变换也是应传统的傅里叶变换不能够满足信号处理的要求而产生的。短时傅里叶变换分析的基本思想是:假定非平稳信号在分析窗函数 $g(t)$ 的一个短时间间隔内是平稳(伪平稳)的,并移动分析窗函数,使 $f(t)g(t-\tau)$ 在不同的有限时间宽度内是平稳信号,从而计算出各个不同时刻的功率谱。但从本质上讲,短时傅里叶变换是一种单一分辨率的信号分析方法,因为它使用一个固定的短时窗函数。因此,短时傅里叶变换在信号分析上还是存在着不可忽略的缺陷。

小波变换是一种信号的时间-尺度分析方法,它具有多分辨率分析的特点,而且在时频两域都具有表征信号局部特征的能力,是一种窗口大小固定不变但形状可改变,时间窗和频率窗都可以改变的时频局部化分析方法,即在低频部分具有较高的频率分辨率,在高频部分具





有较高的时间分辨率和较低的频率分辨率，很适合于探测正常信号中夹带的瞬态反常现象并展示其成分，所以被誉为分析信号的显微镜，利用连续小波变换进行动态系统故障检测与诊断具有良好的效果。

### 4.2.1 傅里叶变换

在信号处理中重要方法之一是傅里叶变换 (FoMierTrMsroM)，它架起了时间域和频率域之间的桥梁。

对很多信号来说，傅里叶分析非常有用。因为它能给出信号所包含的各种频率成分。但是，傅里叶变换有严重的缺点：变换之后使信号失去了时间信息，它不能告诉人们在某段时间里发生了什么变化。而很多信号都包含有人们感兴趣的非稳态（或者瞬变）特性，如漂移、趋势项、突然变化以及信号的开始或结束。这些特性是信号的最重要部分。因此傅里叶变换不适于分析处理这类信号。

虽然傅里叶变换能够将信号的时域特征和频域特征联系起来，能分别从信号的时域和频域观察，但却不能把二者有机地结合起来。这是因为信号的时域波形中不包含任何频域信息。而其傅里叶谱是信号的统计特性，从其表达式中也可以看出，它是整个时间域内的积分，没有局部化分析信号的功能，完全不具备时域信息。也就是说，对于傅里叶谱中的某一频率，不知道这个频率是在什么时候产生的。这样在信号分析中就面临一对最基本的矛盾：时域和频域的局部化矛盾。

在实际的信号处理过程中，尤其是对非平稳信号的处理中，信号在任一时刻附近的频域特征都很重要。如柴油机缸盖表面的震动信号就是由撞击或冲击产生的，是一瞬变信号，仅从时域或频域上来分析是不够的。这就促使去寻找一种新方法，能够将时域和频域结合起来描述观察信号的时频联合特征，构成信号的时频谱。这就是所谓的时频分析法，也称为时频局部化方法。

### 4.2.2 短时傅里叶变换

由于标准傅里叶变换只在频域里有局部分析的能力，而在时域里不存在这种能力。Dennis Gabor 于 1946 年引入了短时傅里叶变换。短时傅里叶变换的基本思想是：把信号划分成许多小的时间间隔，用傅里叶变换分析每个时间间隔，以便确定该时间间隔存在的频率。其表达式为

$$S(\omega, \tau) = \int_{-\infty}^{\infty} f(t) g^*(\omega - \tau) e^{-j\omega t} dt \quad (4-1)$$

式中，\*表示复共轭； $g(t)$  是有紧支集的函数； $f(t)$  是进入分析的信号。在这个变换中， $e^{j\omega t}$  起着频限的作用， $g(t)$  起着时限的作用。随着时间  $\tau$  的变化， $g(t)$  所确定的“时间窗”在  $t$  轴上移动，使  $f(t)$  “逐渐”进行分析。因此， $g(t)$  往往被称为窗口函数。 $S(\omega, \tau)$  大致反映了  $f(t)$  在时刻  $\tau$  时、频率为  $\omega$  的“信号成分”的相对含量。这样，信号在窗函数上的展开就可以表示为在  $[\tau - \delta, \tau + \delta]$ 、 $[\omega - \varepsilon, \omega + \varepsilon]$  这一区域内的状态，并把这一区域称为窗口， $\delta$  和  $\varepsilon$  分别称为窗口的时宽和频宽，表示了时频分析中的分辨率，窗口越小则分辨率越高。很显然，



希望  $\delta$  和  $\varepsilon$  都非常小, 以便有更好的时频分析效果, 但海森堡测不准原理指出  $\delta$  和  $\varepsilon$  是互相制约的, 两者不可能同时都任意小。(事实上,  $\delta\varepsilon \geq 1/2$ , 且仅当  $g(t) = \frac{1}{\delta\pi^{1/4}} e^{-t^2/(2\delta^2)}$  为高斯函数时, 等号成立。)

由此可见, 短时傅里叶变换虽然在一定程度上克服了标准傅里叶不具有局部分析能力的缺陷, 但它也存在着自身不可克服的缺陷, 即当窗函数  $g(t)$  确定后, 矩形窗口的形状就确定了,  $\tau$ 、 $\omega$  只能改变窗口在相平面上的位置, 而不能改变窗口的形状。可以说, 短时傅里叶变换实质上是具有单一分辨率的分析, 若要改变分辨率, 则必须重新选择窗函数  $g(t)$ 。因此, 短时傅里叶变换用来分析平稳信号尚可, 但对非平稳信号, 在信号波形变化剧烈的时刻, 主频是高频, 要求有较高的时间分辨率 ( $\delta$  要小); 而波形变化比较平缓的时刻, 主频是低频, 则要求有较高的频率分辨率 ( $\varepsilon$  要小)。而短时傅里叶变换不能兼顾两者。

### 4.2.3 小波变换

小波变换提出了变化的时间窗, 当需要精确的低频信息时, 采用长的时间窗; 当需要精确的高频信息时, 采用短的时间窗。

小波变换用的不是时间-频率域, 而是时间-尺度域, 尺度越大, 采用越大的时间窗; 尺度越小, 采用越短的时间窗, 即尺度与频率成反比。

#### 1. 一维连续小波变换

定义 2 设  $\psi(t) \in L^2(R)$ , 其傅里叶变换为  $\hat{\psi}(\omega)$ , 当  $\hat{\psi}(\omega)$  满足允许条件 (完全重构条件或恒等分辨条件)

$$C_\psi = \int_R \frac{|\hat{\psi}(\omega)|^2}{|\omega|} d\omega < \infty \quad (4-2)$$

时, 我们称  $\psi(t)$  为一个基本小波或母小波。将母函数  $\psi(t)$  经伸缩和平移后得

$$\psi_{a,b}(t) = \frac{1}{\sqrt{|a|}} \psi\left(\frac{t-b}{a}\right) \quad a, b \in R; a \neq 0 \quad (4-3)$$

称其为一个小小波序列。其中,  $a$  为伸缩因子;  $b$  为平移因子。对于任意函数  $f(t) \in L^2(R)$  的连续小波变换为

$$W_f(a,b) = \langle f, \psi_{a,b} \rangle = |a|^{-1/2} \int_R f(t) \psi\left(\frac{t-b}{a}\right) dt \quad (4-4)$$

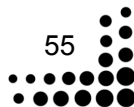
其重构公式 (逆变换) 为

$$f(t) = \frac{1}{C_\psi} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{1}{a^2} W_f(a,b) \psi\left(\frac{t-b}{a}\right) da db \quad (4-5)$$

由于基小波  $\psi(t)$  生成的小波  $\psi_{a,b}(t)$  在小波变换中对被分析的信号起着观测窗的作用, 所以  $\psi(t)$  还应满足一般函数的约束条件

$$\int_{-\infty}^{\infty} |\psi(t)| dt < \infty \quad (4-6)$$

故  $\hat{\psi}(\omega)$  是一个连续函数。这意味着, 为了满足完全重构条件式,  $\hat{\psi}(\omega)$  在必须等于 0, 即







$$\hat{\psi}(0) = \int_{-\infty}^{\infty} \psi(t) dt = 0 \quad (4-7)$$

为了使信号重构的实现在数值上是稳定的，处理完全重构条件外，还要求小波  $\psi(t)$  的傅里叶变化满足下面的稳定性条件：

$$A \leq \sum_{j=-\infty}^{\infty} |\hat{\psi}(2^{-j}\omega)|^2 \leq B \quad (4-8)$$

式中， $0 < A \leq B < \infty$ 。

从稳定性条件可以引出一个重要的概念。

定义 3（对偶小波） 若小波  $\psi(t)$  满足稳定性条件式（4-8），则定义一个对偶小波  $\tilde{\psi}(t)$ ，其傅里叶变换  $\hat{\tilde{\psi}}(\omega)$  由下式给出：

$$\hat{\tilde{\psi}}(\omega) = \frac{\hat{\psi}^*(\omega)}{\sum_{j=-\infty}^{\infty} |\hat{\psi}(2^{-j}\omega)|^2} \quad (4-9)$$

注意：稳定性条件式（4-8）实际上是对式（4-9）分母的约束条件，它的作用是保证对偶小波的傅里叶变换存在的稳定性。值得指出的是，一个小波的对偶小波一般不是唯一的，然而在实际应用中，我们又总是希望它们是唯一对应的。因此，寻找具有唯一对偶小波的合适小波也就成为小波分析中最基本的问题。

连续小波变换具有以下重要性质。

（1）线性性：一个多分量信号的小波变换等于各个分量的小波变换之和。

（2）平移不变性：若  $f(t)$  的小波变换为  $W_f(a, b)$ ，则  $f(t - \tau)$  的小波变换为  $W_f(a, b - \tau)$ 。

（3）伸缩共变性：若  $f(t)$  的小波变换为  $W_f(a, b)$ ，则  $f(ct)$  的小波变换为  $\frac{1}{\sqrt{c}} W_f(ca, cb)$ ， $c > 0$ 。

（4）自相似性：对应不同尺度参数  $a$  和不同平移参数  $b$  的连续小波变换之间是自相似的。

（5）冗余性：连续小波变换中存在信息表述的冗余度。

小波变换的冗余性事实上也是自相似性的直接反映，它主要表现在以下两个方面：

（1）由连续小波变换恢复原信号的重构分式不是唯一的。也就是说，信号  $f(t)$  的小波变换与小波重构不存在一一对应关系，而傅里叶变换与傅里叶反变换是一一对应的。

（2）小波变换的核函数即小波函数  $\psi_{a,b}(t)$  存在许多可能的选择（例如，它们可以是非正交小波、正交小波、双正交小波，甚至允许是彼此线性相关的）。

小波变换在不同的  $(a, b)$  之间的相关性增加了分析和解释小波变换结果的困难，因此，小波变换的冗余度应尽可能减小，它是小波分析中的主要问题之一。

## 2. 高维连续小波变换

对  $f(t) \in L^2(R^n) (n > 1)$ ，公式

$$f(t) = \frac{1}{C_\psi} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{1}{a^n} W_f(a, b) \psi\left(\frac{t-b}{a}\right) da db \quad (4-10)$$

存在几种扩展的可能性，一种可能性是选择小波  $f(t) \in L^2(R^n)$  使其为球对称，其傅里叶变换也同样球对称，即



$$\hat{\psi}(\bar{\omega}) = \eta(|\bar{\omega}|) \quad (4-11)$$

并且其相容性条件变为

$$C_{\psi} = (2\pi)^2 \int_0^\infty |\eta(t)|^2 \frac{dt}{t} < \quad (4-12)$$

对所有的  $f$ , 有  $g \in L^2(g^n)$ 。

$$\int_0^\infty \frac{da}{a^{n+1}} W_f(a, b) \bar{W}_g(a, b) db = C_{\psi} < f \quad (4-13)$$

式中,  $W_f(a, b) = \psi^{a, b}$ ;  $\psi^{a, b}(t) = a^{-n/2} \psi(\frac{t-b}{a})$ 。其中,  $a \in R^+$ ,  $a \neq 0$  且  $b \in R^n$ , 式(4-10)

也可以写为

$$f = C_{\psi}^{-1} \int_0^\infty \frac{da}{a^{n+1}} \int_{R^n} W_f(a, b) \psi^{a, b} db \quad (4-14)$$

如果选择的小波  $\psi$  不是球对称的, 则可以用旋转进行同样的扩展与平移。例如, 在二维时, 可定义

$$\psi^{a, b, \theta}(t) = a^{-1} \psi \left[ (R_{\theta}^{-1} \frac{t-b}{a}) \right] \quad (4-15)$$

这里,  $a > 0, b \in R^2$ ,  $R_{\theta} = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix}$ , 相容条件变为

$$C_{\psi} = (2\pi)^2 \int_0^\infty \frac{dr}{r} \int_0^{2\pi} |\hat{\psi}(r \cos \theta, r \sin \theta)|^2 d\theta < \quad (4-16)$$

该等式对应的重构公式为

$$f = C_{\psi}^{-1} \int_0^\infty \frac{da}{a^3} \int_{R^2} db \int_0^{2\pi} W_f(a, b, \theta) \psi^{a, b, \theta} d\theta \quad (4-17)$$

对于高于二维的情况, 可以给出类似的结论。

### 3. 离散小波变换

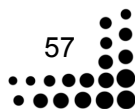
在实际运用中, 尤其是在计算机上实现时, 连续小波必须加以离散化。因此, 有必要讨论连续小波  $\psi_{a, b}(t)$  和连续小波变换  $W_f(a, b)$  的离散化。需要强调的是, 这一离散化都是针对连续的尺度参数  $a$  和连续平移参数  $b$  的, 而不是针对时间变量  $t$  的。这点与我们以前习惯的时间离散化不同。在连续小波中, 考虑函数

$$\psi_{a, b}(t) = |a|^{-1/2} \psi(\frac{t-b}{a})$$

这里  $b \in R$ ,  $a \in R^+$ , 且  $a \neq 0$ ,  $\psi$  是容许的。为方便起见, 在离散化中, 总限制  $a$  只取正值, 这样相容性条件就变为

$$C_{\psi} = \int_0^\infty \frac{|\hat{\psi}(\bar{\omega})|}{|\bar{\omega}|} d\bar{\omega} < \quad (4-18)$$

通常, 把连续小波变换中尺度参数  $a$  和平移参数  $b$  的离散公式分别取作  $a = a_0^j, b = ka_0^j b_0$ , 这里  $j \in Z$ 。扩展步长  $a_0 \neq 1$  是固定值, 为方便起见, 总是假定  $a_0 > 1$  (由于  $m$  可取正也可取负, 所以这个假定无关紧要)。所以对应的离散小波函数  $\psi_{j, k}(t)$  即可写作





$$\psi_{j,k}(t) = a_0^{-j/2} \psi\left(\frac{t - ka_0^j b_0}{a_0^j}\right) = a_0^{-j/2} \psi(a_0^{-j} t - kb_0) \quad (4-19)$$

而离散化小波变换系数则可表示为

$$C_{j,k} = \int_{-\infty}^{\infty} f(t) \psi_{j,k}^*(t) dt = \langle f, \psi_{j,k} \rangle \quad (4-20)$$

其重构公式为

$$f(t) = C \sum_{j=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} C_{j,k} \psi_{j,k}(t) \quad (4-21)$$

式中,  $C$  是一个与信号无关的常数。然而, 怎样选择  $a_0$  和  $b_0$  才能够保证重构信号的精度呢? 显然, 网格点应尽可能密 ( $a_0$  和  $b_0$  尽可能小), 因为网格点越稀疏, 使用的小波函数  $\psi_{j,k}(t)$  和离散小波系数  $C_{j,k}$  就越少, 信号重构的精确度也就会越低。

实际计算中不可能对全部尺度因子值和位移参数值计算 CWT ( $a, b$ ) 值, 加之实际的观测信号都是离散的, 所以信号处理中都是用离散小波变换 (DWT)。大多数情况下是将尺度因子和位移参数按 2 的幂次进行离散。最有效的计算方法是 S. Mallat 于 1988 年发展的快小波算法 (又称塔式算法)。对任一信号, 离散小波变换第一步运算是将信号分为低频部分 (称为近似部分) 和离散部分 (称为细节部分)。近似部分代表了信号的主要特征。第二步对低频部分再进行相似运算, 不过这时尺度因子已经改变。依次进行到所需要的尺度。除了连续小波 (CWT)、离散小波 (DWT), 还有小波包 (Wavelet Packet) 和多维小波。

## 4.3 基于 MATLAB 的小波快速算法设计

Mallat 以多分辨分析为基础提出了著名的快速小波算法——Mallat 算法。该算法在小波变换中的地位如同快速傅里叶变换在傅里叶变换中的地位一样, 可以大大减少小波变换的计算量, 使小波变换真正成为继傅里叶变换后, 处理非平稳信号的有力工具。MATLAB 是一种面向科学与工程计算的高级语言。利用 MATLAB 的强大功能可完成对 Mallat 算法设计的实现。

小波分析是近 20 多年发展起来的新兴学科, 是目前国际上公认的信号信息获取与处理领域的高新技术, 是多学科共同关注的热点, 是信号处理的前沿课题。MATLAB 是一种面向科学与工程计算的高级语言, 由于其强大的功能和广泛的应用性, 受到越来越多科技工作者的欢迎。MATLAB 是科学计算、数值仿真及数据可视化的重要工具。

### 4.3.1 小波快速算法设计原理与步骤

在小波的分解中, 必须使用两个序列  $\{a_n\}$ 、 $\{b_n\}$ 。在正交小波时, 它们是非对称的; 在双正交时, 一般总是要求它们是对称的, 这时  $\{a_n\}$ 、 $\{b_n\}$  还是有限长的。

先讨论一般情况, 即非对称的情形。为了讨论方便, 假定  $\{a_n\}$ 、 $\{b_n\}$ ,  $-L \leq n \leq 0$ , 即  $n < -L$  与  $n > 0$  时,  $\{a_n\}$ 、 $\{b_n\}$  等于零。令  $N$  是我们选择的信号采样的水平, 对于某个正整数  $M$ , 希望分解信号从水平  $N$  到水平  $N - M$ 。



在实际计算中, 给出的初始数据都是有限个。假设给定的数据  $\{c_{N,n}\}$ ,  $-L_1 \leq n \leq L_2$ , 其中  $L_1, L_2$  是两个正整数, 计算  $\{d_{k,n}\}$  与  $\{c_{N-M,n}\}$ ,  $k = N-1, N-2, \dots, N-M$ ;  $-L_1 \leq n \leq L_2$ 。

### 4.3.2 小波分解算法

输入: 正整数  $N, M, L, L_1, L_2$ ;

输入数据  $\{c_{N,n}\}$ ,  $-L_1 \leq n \leq L_2$ ;

预先存储的序列  $\{a_n\}, \{b_n\}$ ,  $-L \leq n \leq 0$ 。

输出: 序列  $\{d_{k,n}\}, \{c_{N-M,n}\}$ ,  $N-M \leq k \leq N-1, -L_1 \leq n \leq L_2$ 。

步骤 1: 计算  $\{L_1^{(k)}\}, \{L_2^{(k)}\}$ ,  $k = N-1, N-2, \dots, N-M$ 。

使用公式:

$$k = N-1, N-2, \dots, N-M$$

$$L_1^{(N-1)} = \lfloor L_1 / 2 \rfloor, \quad L_1^{(k)} = \left\lfloor \frac{L_1^{(k-1)}}{2} \right\rfloor$$

$$L_2^{(N-1)} = \lfloor L_2 / 2 \rfloor, \quad L_2^{(k)} = \left\lfloor \frac{L + L_2^{(k-1)}}{2} \right\rfloor$$

步骤 2: 计算  $\{c_{k,n/2}\}$  与  $\{d_{k,n/2}\}$ ,  $k = N-1, N-2, \dots, N-M$

使用公式:

$$c_{k,n/2} = \sum_{j=-L+n}^n a_{j-n} c_{k+1,j}$$

$$d_{k,n/2} = \sum_{j=-L+n}^n b_{j-n} c_{k+1,j}$$

式中,  $n$  是偶数, 并且  $-L_1^{(k+1)} \leq n \leq L_2^{(k+1)}$ , 即要求  $-L_1^{(k)} \leq \lfloor n/2 \rfloor \leq L_2^{(k)}$ 。

计算复杂性: 所需要的算术运算总次数不超过  $4(L+1)(L_1 + L_2 + ML - 2L + M)$ 。

### 4.3.3 对称小波分解算法

现在讨论  $\{a_n\}, \{b_n\}$  是对称的情形。通过指标平移, 总能够假定  $a_{-n} = a_n, b_{-n} = b_n$ 。假定对于正整数  $L_3, L_4$ , 使

$$\{a_n\}, -L_3 \leq n \leq L_3; \{b_n\}, -L_4 \leq n \leq L_4$$

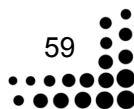
这样, 我们有下述对称小波分解算法。

输入: 正整数  $N, M, L_1, L_2, L_3, L_4$ ;

输入数据  $\{c_{N,n}\}$ ,  $-L_1 \leq n \leq L_2$ ;

预先存储的序列  $\{a_n\}, -L_3 \leq n \leq 0; \{b_n\}, -L_4 \leq n \leq 0$ 。

输出: 序列  $\{d_{k,n}\}, \{c_{N-M,n}\}$ ,  $N-M \leq k \leq N-1, -L_1 \leq n \leq L_2$ 。





步骤 1: 同 4.3.2 小节的算法步骤 1。

步骤 2: 计算  $\{c_{k,n/2}\}$  与  $\{d_{k,n/2}\}$ ,  $k=N-1, N-2, \dots, N-M$

使用公式:

$$c_{k,n/2} = a_0 c_{k+1,n} + \sum_{j=L_3+n}^{n-1} a_{j-n} (c_{k+1,j} + c_{k+1,2n-j})$$

$$d_{k,n/2} = b_0 c_{k+1,n} + \sum_{j=-L_4+n}^{n-1} b_{j-n} (c_{k+1,j} + c_{k+1,2n-j})$$

式中,  $n$  是偶数, 并且  $-L_1^{(k+1)} \leq n \leq L_2^{(k+1)}$ 。

公式推导: 由分解算法公式可得

$$\begin{aligned} c_{k,n/2} &= \sum_{l=-L_3+n}^{L_3+n} a_{l-n} c_{k+1,l} \\ &= a_0 c_{k+1,n} + \sum_{l=-L_3+n}^{n-1} a_{l-n} c_{k+1,l} + \sum_{l=n+1}^{L_3+n} a_{l-n} c_{k+1,l} \\ &= a_0 c_{k+1,n} + \sum_{l=-L_3+n}^{n-1} a_{l-n} c_{k+1,l} + \sum_{j=n-1}^{-L_3+n} a_{n-j} c_{k+1,2n-j} \\ &= a_0 c_{k+1,n} + \sum_{l=-L_3+n}^{n-1} a_{l-n} (c_{k+1,l} + c_{k+1,2n-l}) \end{aligned}$$

$d_{k,n/2}$  的推导与其类似。

计算复杂性: 所需要的算术运算次数不超过  $5(L+1)(L_1 + L_2 + ML - 2L + M)$ , 式中,  $L = \max\{L_3, L_4\}$ 。

#### 4.3.4 小波重构算法

下面讨论重构算法。对于信号的重构, 需要使用两个序列  $\{p_n\}$ 、 $\{q_n\}$ 。仍然先看一般情形, 即  $\{p_n\}$ 、 $\{q_n\}$  非对称的情况。为了方便, 对于正整数  $r$ 、 $s$ , 两个有限序列分别是  $\{p_n\}$ ,  $0 \leq n \leq r-1$ ;  $\{q_n\}$ ,  $0 \leq n \leq s-1$ 。同样,  $\{p_n\}$ 、 $\{q_n\}$  是预先存储的。

由于信号重构是分解的逆过程, 这时输入信号  $\{c_{N-M,i}\}$  与序列  $\{d_{k,i}\}$ ,  $k = N-M, \dots, N-1$ ,  $-L_1^{(k)} \leq i \leq L_2^{(k)}$ 。对  $k = N-M, \dots, N-1$  和  $-L_1^{(k)} \leq i \leq L_2^{(k)}$  使用公式

$$\begin{aligned} c_{k+1,i} &= \sum_{j=i}^{i+r} p_{i-j} c_{k,j/2} + \sum_{j=i}^{i+s} q_{i-j} d_{k,j/2} \\ &= \sum_{j=i}^{i+\max\{r,s\}} (p_{i-j} c_{k,j/2} + q_{i-j} d_{k,j/2}) \end{aligned}$$

式中,  $j$  为奇数时,  $c_{k,j/2} = d_{k,j/2} = 0$ 。

输入: 正整数  $N, M, r, s, L_1, L_2$ ;

序列  $\{c_{N-M,i}\}$ 、 $\{d_{k,i}\}$ ,  $N-M \leq k \leq N-1$ ,  $-L_1^{(k)} \leq i \leq L_2^{(k)}$ ;

预先存储的序列  $\{p_i\}$ ,  $0 \leq i \leq r-1$ ,  $\{q_i\}$ ,  $0 \leq i \leq s-1$ 。



输出：序列  $\{c_{N,i}\}$  ,  $-L_1 \leq i \leq L_2$ 。

步骤：计算  $\{c_{k,j}\}$  ,  $k = N - M + 1, n - M + 2, \dots, N$ 。

使用公式：

$$c_{k+1,i} = \sum_{j=i}^{i+L} (p_{i-j} c_{k,j/2} + q_{i-j} d_{k,j/2})$$

式中,  $-L_1^{(k)} \leq n \leq L_2^{(k)}$  , 且  $j$  为奇数时,  $c_{k,j/2} = d_{k,j/2} = 0$  , 而  $L = \max\{r, s\}$ 。

这种重构算法与分解算法的算术运算次数基本相同。

### 4.3.5 对称小波重构算法

下面讨论  $\{p_n\}$ 、 $\{q_n\}$  对称的情况。通过平移, 总可假定

$$p_{-n} = p_n, \quad q_{-n} = q_n$$

而且

$$\{p_n\}, -r \leq n \leq r; \{q_n\}, -s \leq n \leq s$$

输入：正整数  $N, M, r, s, L_1, L_2$  ;

序列  $\{c_{N-M,n}\}, \{d_{k,n}\}, N - M \leq k \leq N - 1, -L_1^{(k)} \leq n \leq L_2^{(k)}$  ;

预先存储的序列  $\{p_n\}, 0 \leq n \leq r, \{q_n\}, 0 \leq n \leq s$ 。

输出：序列  $\{c_{N,n}\}$  ,  $-L_1 \leq n \leq L_2$ 。

步骤：计算  $\{c_{k,i}\}$  ,  $k = N - M + 1, n - M + 2, \dots, N$ 。

使用公式：

$$c_{k+1,i} = p_0 c_{k,i/2} + q_0 d_{k,i/2} + \sum_{j=i+1}^{i+L} [p_{i-j} (c_{k,j/2} + c_{k,i-j/2}) + q_{i-j} (d_{k,j/2} + d_{k,i-j/2})]$$

式中,  $-L_1^{(k)} \leq n \leq L_2^{(k)}$  , 且  $j$  为奇数时,  $c_{k,j/2} = d_{k,j/2} = 0$  , 而  $L = \max\{r, s\}$ 。

上述算法的推导类似于对称小波分解算法的推导。

### 4.3.6 MATLAB 程序设计实现

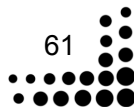
数学作为基础学科, 与工程技术及科学研究领域密不可分, 并且在这些学科中得到了越来越广泛的应用。我们用小波分析这一数学工具处理一些数学问题, 从某种意义上讲, 是帮助读者进一步理解小波分析理论本身。

【例 4-1】利用 MATLAB 实现小波分解与重构快速算法。对图 4-1 (a) 所示的信号进行分解, 其具体过程如下。

(1) 输入原始信号并显示

利用 MATLAB 中的随机函数产生原始信号, 并且显示该信号。

```
% 使用随机函数产生一维信号
clc;clear
```





```
wname='db3';  
randn('seed',531316785);  
s=2+kron(ones(1,8),[1 -1])+((1:16).^2)/32+0.2*randn(1,16);  
figure;  
plot(s);title('(a) 原始图像');
```

### (2) 对原始信号利用 db3 小波进行分解

利用 MATLAB 中的小波分解函数对前面产生的原始信号进行分解,如图 4-1 (b) ~ 图 4-1 (e) 所示。

```
% 利用 db3 小波对原始信号进行二尺度分解  
% 使用小波分解函数 dwt  
[cA1,cD1]=dwt(s,wname);  
figure;plot(cA1);title('(b) 近似系数 cA1');  
figure;plot(cD1);title('(c) 细节系数 cD1');  
% 对一尺度上的近似系数再次进行小波分解  
[cA2,cD2]=dwt(cA1,wname);  
figure;plot(cA2);title('(d) 近似系数 cA2');  
figure;plot(cD2);title('(e) 细节系数 cD2');
```

### (3) 由分解信号重构原始信号

利用 MATLAB 中的小波重构函数对分解的小波信号进行重构,如图 4-1 (f) 所示。

```
[Lo_R,Hi_R]=wfilters(wname,'r');  
ss=idwt(cA2,cD2,Lo_R,Hi_R);  
sss=idwt(ss,cD1,Lo_R,Hi_R);  
% 计算机重构信号与原始信号的误差  
err=norm(s-sss)  
figure;plot(sss);title('(f) 重构的原始信号');  
xlabel(['相对误差=',num2str(err)]);
```

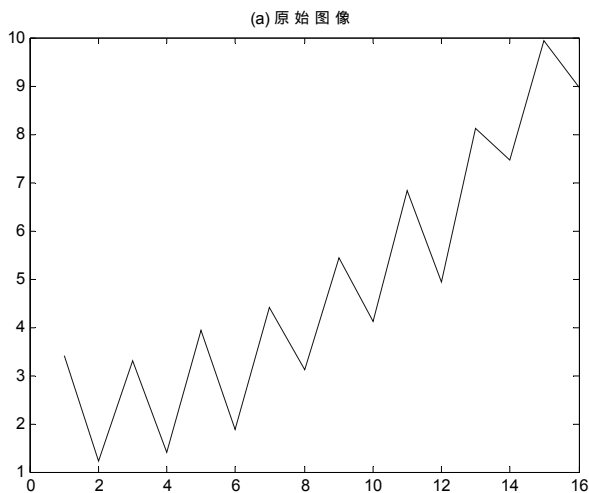


图 4-1 小波分解与重构示意图

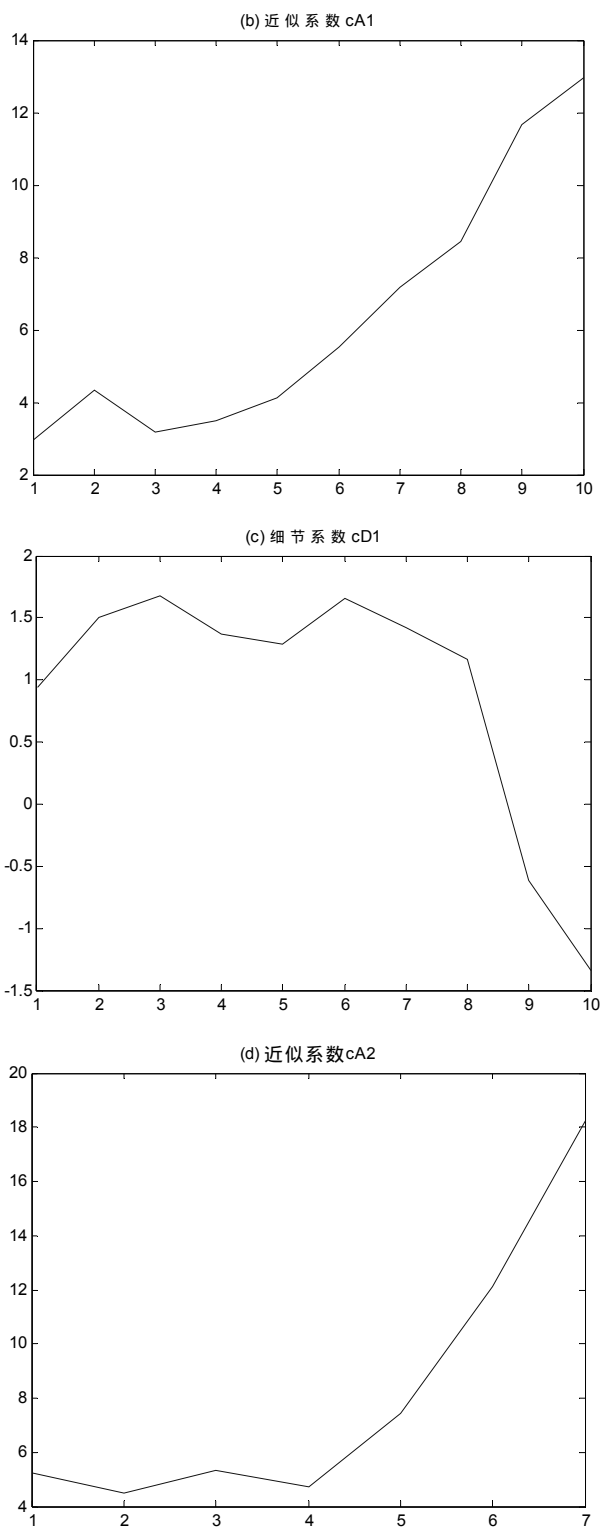


图 4-1 小波分解与重构示意图 (续一)



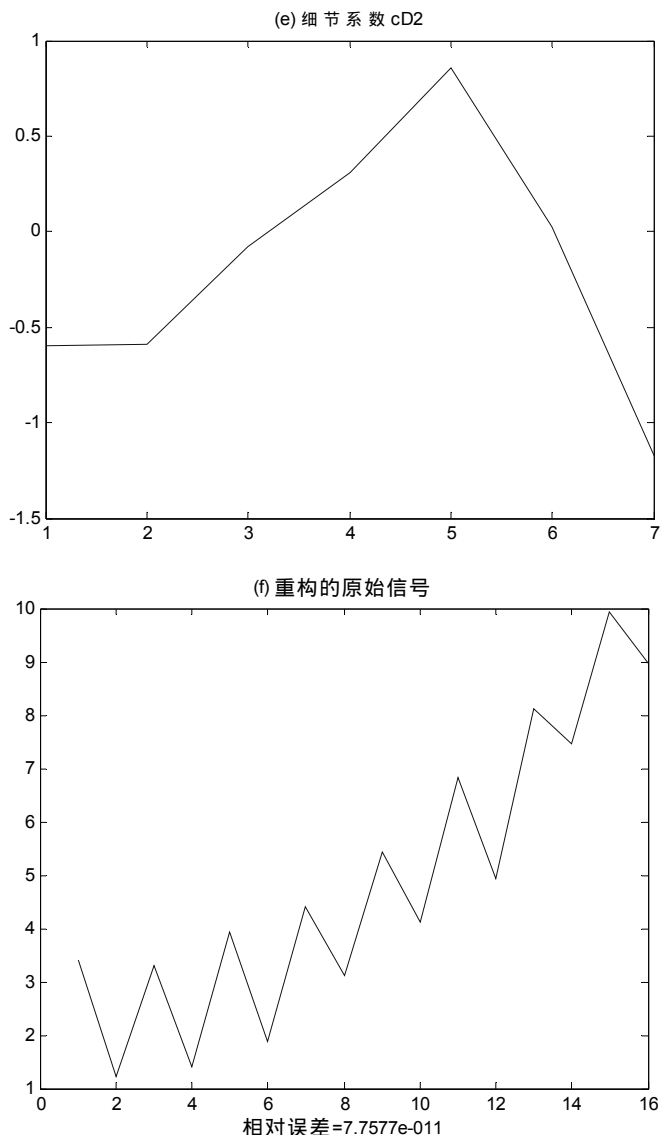


图 4-1 小波分解与重构示意图 (续二)

多级小波分解通过级联的方式进行，每级的小波变换都是在前一级分解产生的低频分量上的继续，重构是分解的逆运算。低频分量上的信息比较丰富，能量集中；高频分量上的信息分量多为零，细节信息丰富，能量较少。

Mallat 算法通过一组分解低通滤波器和分解高通滤波器对信号进行滤波，然后对输出结果进行下二采样（指隔一取一）来实现小波分解，分解的结果是产生长度减半的两部分，一个是经低通滤波器产生的原始信号的平滑部分，另一个则是经高通滤波器产生的原始信号细节部分。重构时使用一组重构低通滤波器和重构高通滤波器对小波分解的结果滤波，再进行上二采样（相邻两点间补零）来生成重构信号。

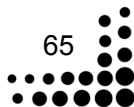
【例 4-2】对于一给定的正弦信号  $s(i) = \sin(i\pi/100 + \pi/4)$ ,  $i = 0, 1, \dots, 199$ ，请利用多分辨分析对该信号进行分解与重构。



该问题可以说是一个纯粹的数学问题，通过对该问题的讲解，可以加深对小波分析中多分辨分析的理解，即如何对信号进行多层分解与重构。

在这里，我们分别选用 db1 和 coif3 小波对该正弦信号进行 3 层多分辨分析，处理过程可编程如下：

```
t=0:pi/100:4*pi;
s=sin(t+pi/4);
subplot(532);plot(s);
title('原始信号');
[c,l]=wavedec(s,3,'db1');grid;
ca3=appcoef(c,l,'db1',3);           %提取小波分解的低频系数
cd3=detcoef(c,l,3);                 %提取第三层的高频系数
cd2=detcoef(c,l,2);                 %提取第二层的高频系数
cd1=detcoef(c,l,1);                 %提取第一层的高频系数
figure(2);
subplot(421);plot(ca3);
title(' db1 第三层低频系数');
subplot(423);plot(cd1);
title(' db1 第一层高频系数');
subplot(425);plot(cd2);
title(' db1 第二层高频系数');
subplot(427);plot(cd3);
title(' db1 第三层高频系数');
[c,l]=wavedec(s,3,'coif3');grid;
s1=waverec(c,l,'coif3')
ca3=appcoef(c,l,'coif3',3);         %提取小波分解的低频系数
cd3=detcoef(c,l,3);                 %提取第三层的高频系数
cd2=detcoef(c,l,2);                 %提取第二层的高频系数
cd1=detcoef(c,l,1);                 %提取第一层的高频系数
subplot(422);plot(ca3);
title(' coif3 第三层低频系数');
subplot(424);plot(cd1);
title(' coif3 第一层高频系数');
subplot(426);plot(cd2);
title(' coif3 第二层高频系数');
subplot(428);plot(cd3);
title(' coif3 第三层高频系数');
s2=waverec(c,l,'coif3')
figure(3);
subplot(521);plot(s1);grid;
title('db1 小波重构信号');
[c,l]=wavedec(s,3,'coif3');
```





```
subplot(522);plot(s2);grid;  
title('coif3 小波重构信号');
```

输出结果如图 4-2 所示。

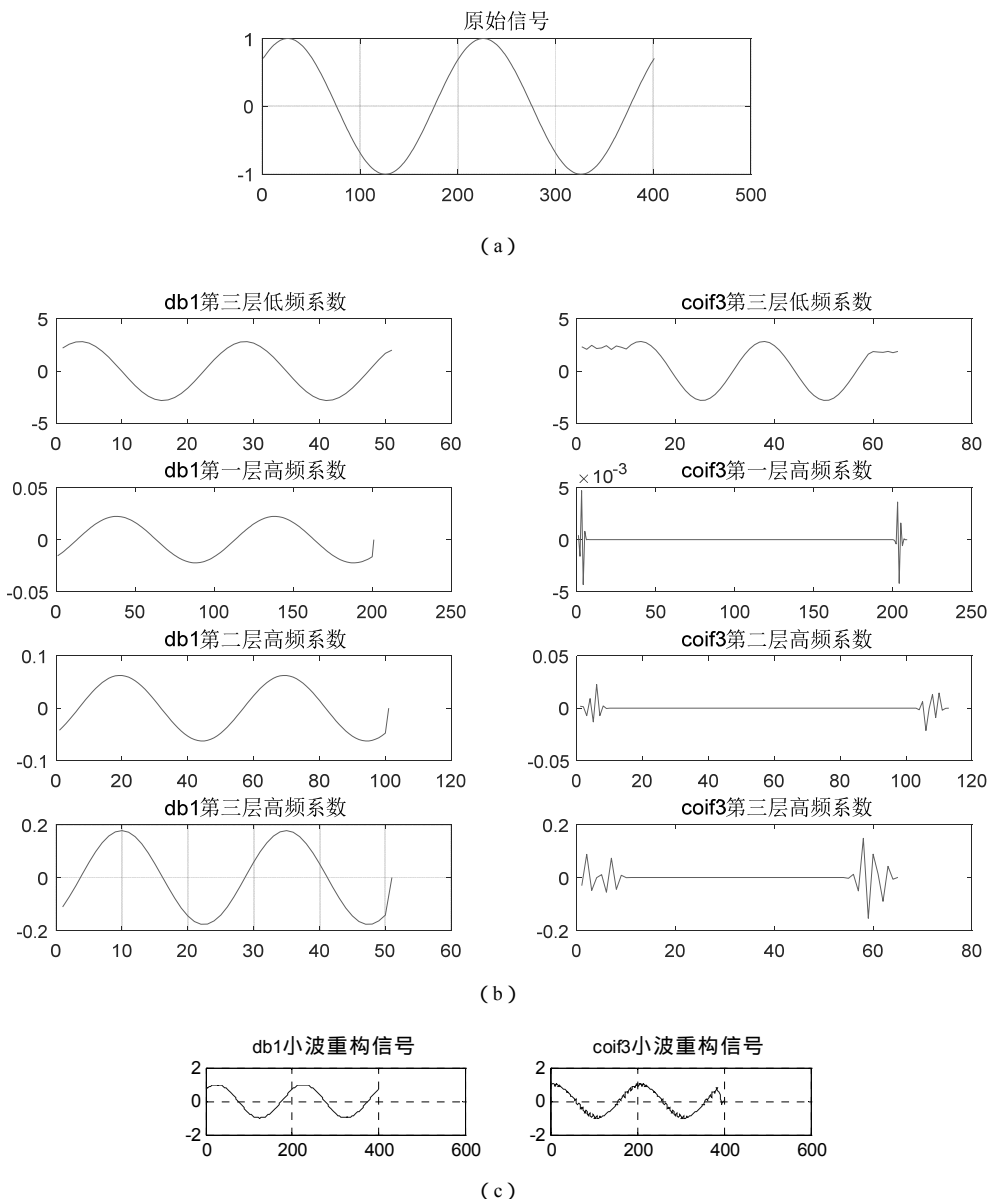


图 4-2 信号分解与重构实例

【例 4-3】给定一信号（文件名为 `leleccum.mat`），请用 `db1` 小波对信号分别进行单尺度和三尺度分解，求出各次分解的低频系数和高频系数，并分别对低频系数、高频系数以及低高频系数进行重构。

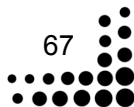
该问题是运用小波分析对信号单、多尺度分解与重构方法的一次全面的讨论。通过该



问题的分析，主要让读者清楚如何用小波分析对信号进行单尺度、多尺度分解以及如何对信号进行部分重构和全面重构。

程序清单如下：

```
% 装载一原始的一维信号
load leleccum;s=leleccum(1:3920);
ls=length(s);
% 画出原始波形图
figure(1);
subplot(511);plot(s);
ylabel('s');
title('原始信号 s 及单尺度分解的低频系数 ca1 和 高频系数 cd1');
% 用小波 db1 进行单尺度一维分解
[ca1,cd1]=dwt(s,'db1');
% 画出分解后的低频系数 ca1 和 高频系数 cd1
subplot(523);plot(ca1);
ylabel('ca1');
subplot(524);plot(cd1);
ylabel('cd1');
% 分别对低频系数 ca1 和 高频系数 cd1 进行重构
a1=upcoef('a',ca1,'db1',1,ls);
d1=upcoef('d',cd1,'db1',1,ls);
% 分别画出重构后低频部分和 高频部分的波形图
figure(2);
subplot(511);plot(a1);
ylabel('a1');
title('单尺度分解的低频重构信号、 高频重构信号及合成重构信号');
subplot(512);plot(d1);
ylabel('d1');
% 画出 a1+d1 波形图，即对 s 的分解系数直接进行重构
a0=idwt(ca1,cd1,'db1',ls);
subplot(513);plot(a0);
ylabel('a1+d1');
% 用 db1 小波对信号进行三层分解
[c,l]=wavedec(s,3,'db1');
% 从小波分解结构[c,l]中提取低频系数
ca3=appcoef(c,l,'db1',3);
% 从小波分解结构[c,l]中提取第 1、2、3 层的高频系数
cd3=detcoef(c,l,3);
cd2=detcoef(c,l,2);
cd1=detcoef(c,l,1);
% 分别画出原始信号、低频系数和 高频系数的波形
figure(3);
```





```
subplot(511);plot(s);
ylabel('s');
title('原始信号及三层分解的各层分解系数');
subplot(589);plot(ca3);
ylabel('ca3');
subplot(5,8,17);plot(cd3);
ylabel('cd3');
subplot(5,4,13);plot(cd2);
ylabel('cd2');
subplot(5,2,9);plot(cd1);
ylabel('cd1');
% 对第 3 层的低频系数进行重构
a3=wrcoef('a',c,l,'db1',3);
% 从小波分解结构[c,l]中提取第 1、2、3 层的高频系数进行重构
d3=wrcoef('d',c,l,'db1',3);
d2=wrcoef('d',c,l,'db1',2);
d1=wrcoef('d',c,l,'db1',1);
% 画出各层系数重构后的波形图
figure(4);
subplot(511);plot(ca3);
ylabel('ca3');
title('各层分解系数的重构图及合成重构图 a0');
subplot(512);plot(cd3);
ylabel('cd3');
subplot(513);plot(cd2);
ylabel('cd2');
subplot(514);plot(cd1);
ylabel('cd1');
% 对小波分解结构[c,l]进行重构
a0=waverec(c,l,'db1');
subplot(515);plot(a0);
ylabel('a0');
```

输出结果如图 4-3 所示。

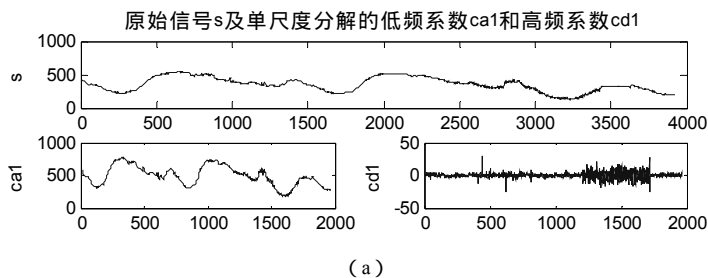
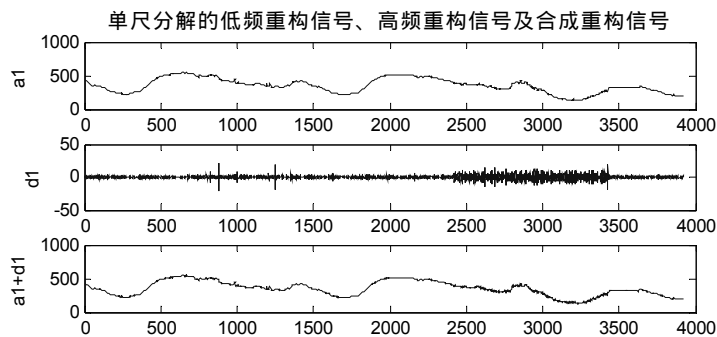
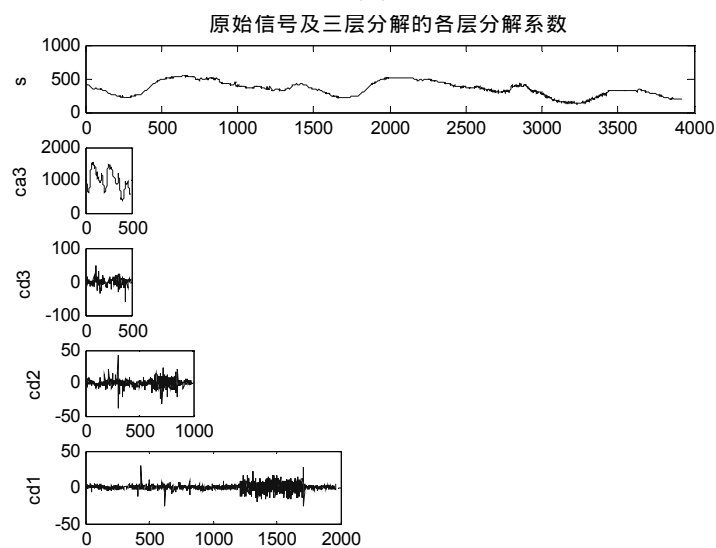


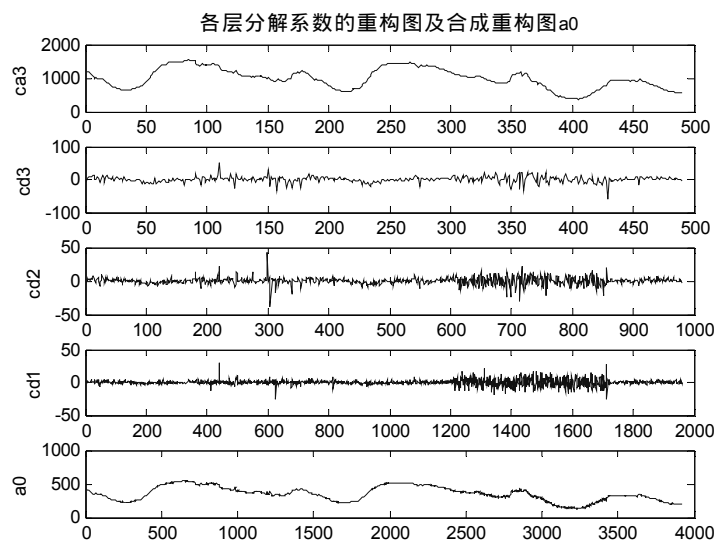
图 4-3 信号的分解与重构实例



(b)

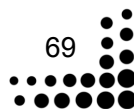


(c)



(d)

图 4-3 信号的分解与重构实例 (续)





从上面的分析我们可以看出，小波分析可以把信号的不同频率区域分开。

【例 4-4】下面给出用傅里叶变换和 db1 小波变换实现的对该信号在频域内的表示形式，从比较中可以更好地领会傅里叶变换在处理时域上有变化的信号的不足之处。

程序清单如下：

```
% 信号的傅里叶变换
load freqbrk;          % 装入要分析的信号
s=freqbrk;
ls=length(s);
subplot(6,1,1);plot(s);title('原始信号的时域图');
% 对信号 s 进行 FFT 变换
fs=fft(s,1024);        % 在 s 信号中取 1024 个点，倘若 s 中不够长，则后面补零
fs=abs(fs); % 将 FFT 后的复数用 abs 求其模的大小，返回的值是复数的模
subplot(6,1,2);plot(fs);
ylabel('FFT');grid;
% 信号用 db1 小波分解到第三层后的频域特性
[c,l]=wavedec(s,3,'db1'); % 用 db1 小波分解信号到第三层
% 对分解结构[c,l]的第三层低频部分进行重构
a3=wrcoef('a',c,l,'db1',3);
subplot(6,1,3);plot(a3);ylabel('a3');
% 对分解结构[c,l]中的各层高频部分进行重构
for i=1:3
    decmp=wrcoef('d',c,l,'db1',4-i);
    subplot(6,1,i+3);
    plot(decmp);
    ylabel(['d',int2str(4-i)]);
end
```

输出结果如图 4-4 所示，可以看出，由于傅里叶变换将信号变换成纯频域信号，使它不具备时间分辨率，故对信号的频率变化点根本无法检测出来，而经 db1 小波分解后的信号，则可以很明显地辨别出该断裂点。

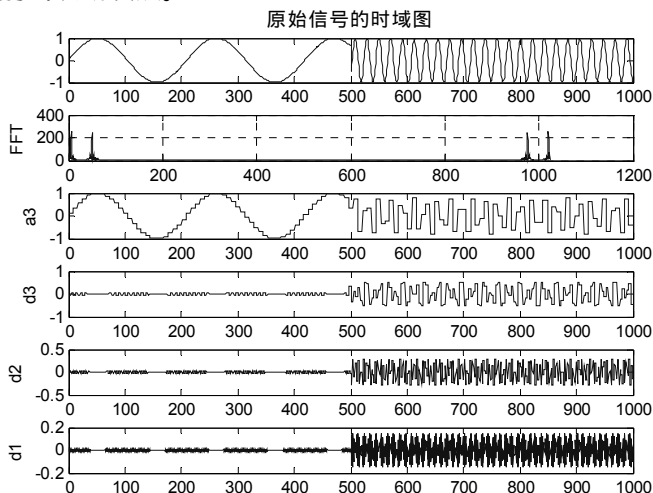


图 4-4 小波变换对信号分析检测



【例 4-5】给定一信号，请利用小波分析把信号的各个频率成分分开。

在傅里叶分析中，如果一个信号是由几个不同频率的正弦信号组成，则变换可以很有效地分辨这些不同频率的正弦信号。现在，我们用小波分析来实现这一功能。所分析的信号由 3 个不同频率的正弦信号组成。

程序清单如下：

```
x=0:0.05:6*pi;
s=sin(x)+sin(10*x)+sin(100*x); % 产生一个正弦叠加信号
figure(1);
subplot(6,2,1);plot(s);
title('原始信号与各层低频部分');
ylabel('s');
subplot(6,2,2);plot(s);
title('原始信号与各层高频部分');
ylabel('s');
[c,l]=wavedec(s,5,'db3'); % 用 db3 小波分解信号到第五层
% 对分解结构[c,l]中各低频部分进行重构，并显示结果
for i=1:5
    decmp=wrcoef('a',c,l,'db3',6-i);
    subplot(6,2,2*i+1);
    plot(decmp);
    ylabel(['a',num2str(6-i)]);
end
% 对分解结构[c,l]中各高频部分进行重构，并显示结果
for i=1:5
    decmp=wrcoef('d',c,l,'db3',6-i);
    subplot(6,2,2*i+2);
    plot(decmp);
    ylabel(['d',num2str(6-i)]);
end
% 画出 d1 的放大波形图
figure(2);
d1=wrcoef('d',c,l,'db3',1);
subplot(411);plot(d1(1:100));
```

程序运行结果如图 4-5 所示。

在图 4-5 显示的分解结果中可以看出，低频的第四层将正弦信号中的最低频率的组成清晰地分离出来。

在小波分解中，若将信号中的最高频率成分看作是 1，则各层小波分解分别是带通或低通滤波器，且各层所占的具体频带为

a1 : 0 ~ 0.5

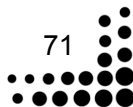
a2 : 0 ~ 0.25

a3 : 0 ~ 0.125

d1 : 0.5 ~ 1

d2 : 0.25 ~ 0.5

d3 : 0.125 ~ 0.25





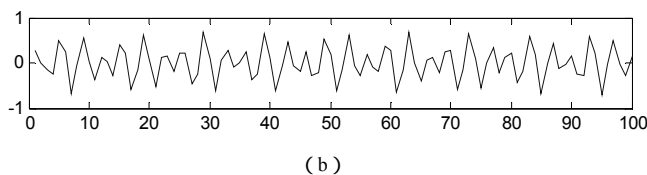
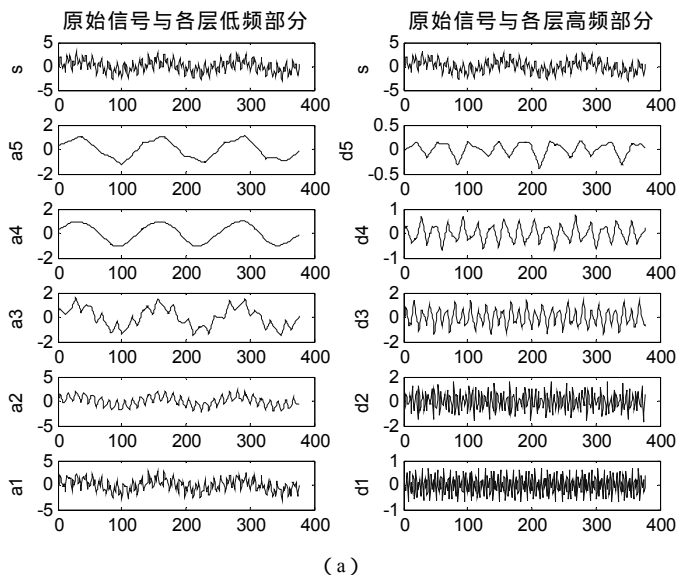
 $a4 : 0 \sim 0.0625$  $d4 : 0.0625 \sim 0.125$  $a5 : 0 \sim 0.03125$  $d5 : 0.03125 \sim 0.0625$ 

图 4-5 用小波分析来分析信号的不同频率

该信号由 3 个不同频率的正弦信号所组成，其频率在小波分解下的相对频率分别为高频正弦（1）、中频正弦（0.1）、低频正弦（0.01）。由信号频率的组成部分和小波分解下各层频率的分布可知，高频正弦一定位于  $d1$  层，实际情况也是如此。图 4-5 (b) 显示了放大后的  $d1$  层信号图，可以看出，每个信号包络中都有 10 个正弦振荡，它的相对周期的估计值为 200（在小波分析中的频率为 1）。 $d4$  层中包含中频层的正弦信号。我们注意到，在  $a3$  层和  $a4$  层中间有不连续的点出现，因为中频正弦信号是由这两层来共同表达的， $d4$  层的信号有一部分被  $d3$  层减去了。我们需要用  $a1 \sim a3$  层的信号来估计中层的正弦信号，将  $a1$  放大后可以看见中频层的正弦信号，其相对周期大约是 20（在小波分析中的频率为 0.1）。最后，只剩下低频正弦信号没有提取出来，在  $a4$  层中对其进行估计，可以看出其相对周期大约是 2，一个周期占 200 个点（在小波分析下的频率为 0.01）。其实，低频正弦信号在  $a5$  层也能看出来，但是，这就必须将信号进行小波的进一步分解。若进一步分解，可以看到，该信号在  $a4$  层消失了，而出现在  $d4$  层。

总之，我们能够用小波分析将合成信号中的单纯正弦信号的频率提取出来。因为在小波分解下，不同的尺度具有不同的时间和频率分辨率，因而小波分解能够将信号的不同频率成分分开。

## 第 5 章 利用小波变换对信号进行分析



### 5.1 信号压缩

#### 5.1.1 信号压缩步骤

对一维信号进行压缩，可以选用小波分析和小波包分析两种手段进行，主要包括以下几个步骤。

步骤 1：信号的小波（或小波包）分解。

步骤 2：对高频系数进行阈值量化处理。对  $1 \sim N$  层的高频系数，均可选择不同的阈值，并且用硬阈值进行系数的量化。

步骤 3：对量化后的系数进行小波（或小波包）重构。

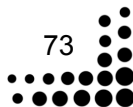
下面给出一个具体的例程，使读者对小波分析在信号压缩中的应用有一个较直观的印象。

#### 5.1.2 信号压缩实例

**【例 5-1】**利用小波分析对给定信号进行压缩处理（一）

使用函数 `wdcbm` 获取信号压缩阈值，然后采用函数 `wdencmp` 实现信号压缩。

```
%装载信号
load nelec;
indx=1:1024;
x=nelec(indx);
% 用小波 Haar 对信号进行 3 层分解
[c,l]=wavedec(x,3,'haar');
alpha=1.5;
% 获取信号压缩的阈值
[thr,nkeep]=wdcbm(c,l,alpha);
% 对信号进行压缩
[xd,cxd,lxd,perf0,perf12]=wdencmp('lvd',c,l,'haar',3,thr,'s');
subplot(2,1,1);
plot(indx,x);
title('原始信号');
```





```
subplot(2,1,2);  
plot(indx,xd);  
title('压缩后的信号');
```

程序运行结果如图 5-1 所示。

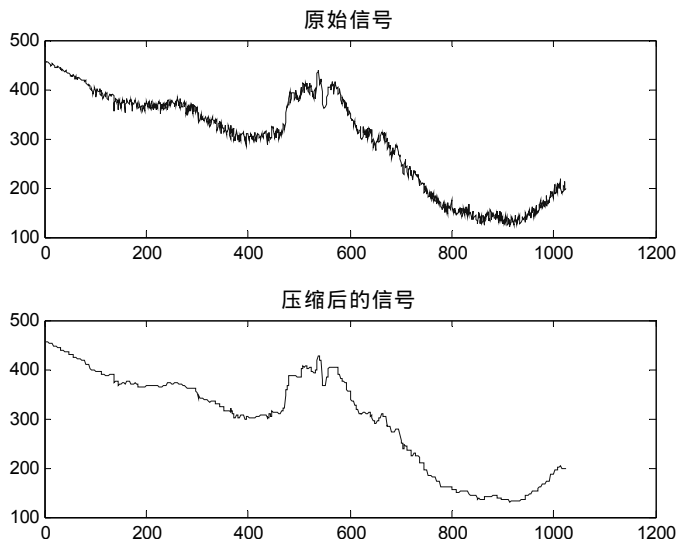


图 5-1 信号压缩结果（一）

### 【例 5-2】利用小波分析对给定信号进行压缩处理（二）

本例使用函数 `ddencmp` 获取信号压缩阈值，然后采用函数 `wdencmp` 实现信号压缩。

```
%装载信号  
load nelec;  
indx=1:1024;  
x=nelec(indx);  
% 用小波 haar 对信号进行五层分解  
[c,l]=wavedec(x,5,'haar');  
% 获取信号压缩的阈值  
[thr,nkeep]=ddencmp('cmp','wv',x);  
% 对信号进行压缩  
xd=wdencmp('gbl',c,l,'haar',5,thr,'s',1);  
subplot(2,1,1);  
plot(indx,x);  
title('原始信号');  
subplot(2,1,2);  
plot(indx,xd);  
title('压缩后的信号');
```

程序运行结果如图 5-2 所示。

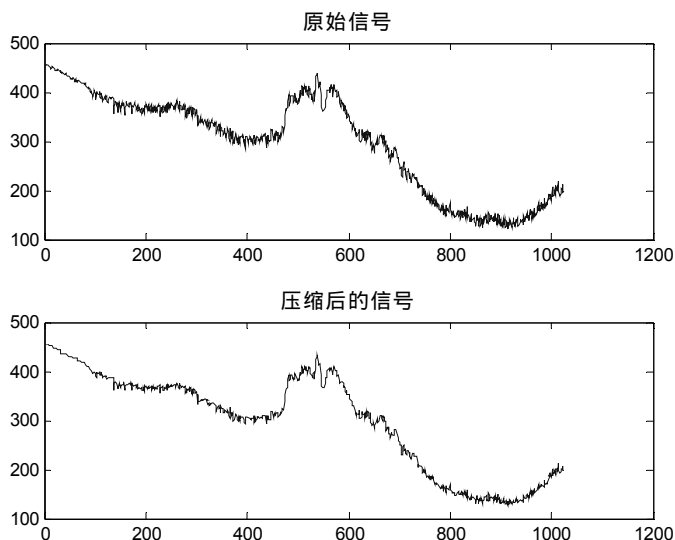


图 5-2 信号压缩结果（二）

信号的压缩与去噪比，主要差别在第二步。一般地，有两种比较有效的信号压缩方法，第一种方法是对信号进行小波尺度的扩展，并且保留绝对值最大的系数。在这种情况下，可以选择使用全局阈值，此时仅需要输入一个参数即可。第二种方法是根据分解后各层的效果来确定某一层的阈值，且每一层的阈值可以是互不相同的。

## 5.2 信号去噪

### 5.2.1 信号去噪步骤

对信号去噪实质上是抑制信号中的无用部分，增强信号中 useful 部分的过程。一般地，一维信号去噪的过程可分为如下 3 个步骤。

步骤 1：一维信号的小波分解。选择一个小波并确定分解的层次，然后进行分解计算。

步骤 2：小波分解高频系数的阈值量化。对各个分解尺度下的高频系数选择一个阈值进行软阈值量化处理。

步骤 3：一维小波重构。根据小波分解的最底层低频系数和各层高频系数进行一维小波重构。

这 3 个步骤中，最关键的是如何选择阈值以及进行阈值量化。在某种程序上，它关系到信号去噪的质量。

总体上，对于一维离散信号来说，其高频部分所影响的是小波分解的第一层细节，其低频部分所影响的是小波分解的最深层和低频层。如果对一个仅由白噪声所组成的信号进行分析，则可得出这样的结论：高频系数的幅值随着分解层次的增加而迅速地衰减，且其方差也有同样的变化趋势。

小波分析工具箱中用于信号去噪的一维小波函数是 `wden.m` 的 `wdencomp.m`。



小波分析进行去噪处理一般有下列 3 种方法。

(1) 默认阈值去噪处理。该方法利用函数 `ddencmp` 生成信号的默认阈值，然后利用函数 `wdencomp` 进行去噪处理。

(2) 给定阈值去噪处理。在实际的去噪过程中，阈值往往可通过经验公式获得，且这种阈值比默认阈值的可信度高。在进行阈值量化处理时可利用函数 `wthresh`。

(3) 强制去噪处理。该方法是将小波分解结构中的高频系数全部置为 0，即滤掉所有高频部分，然后对信号进行小波重构。这种方法比较简单，且去噪后的信号比较平滑，但是容易丢失信号中的有用成分。

## 5.2.2 信号去噪实例

【例 5-3】利用小波分析对污染信号进行去噪处理以恢复原始信号。

```
%装载采集的信号 leleccum.mat
load leleccum;
%将信号中第 1~3920 个采样点赋给 s
s=leleccum(1:3920);
ls=length(s);
%画出原始信号
subplot(2,2,1);
plot(s);
title('原始信号');grid;
%用 db1 小波对原始信号进行 3 层分解并提取系数
[c,l]=wavedec(s,3,'db1');
ca3=appcoef(c,l,'db1',3);
cd3=detcoef(c,l,3);
cd2=detcoef(c,l,2);
cd1=detcoef(c,l,1);
%对信号进行强制性去噪处理并图示结果
cdd3=zeros(1,length(cd3));
cdd2=zeros(1,length(cd2));
cdd1=zeros(1,length(cd1));
c1=[ca3 cdd3 cdd2 cdd1];
s1=waverec(c1,l,'db1');
subplot(2,2,2);
plot(s1);
title('强制去噪后的信号');grid;
%用默认阈值对信号进行去噪处理并图示结果
%用 ddencmp 函数获得信号的默认阈值,使用 wdencomp 命令函数实现去噪过程
[thr,sorh,keepapp]=ddencmp('den','wv',s);
s2=wdencomp('gbl',c,l,'db1',3,thr,sorh,keepapp);
subplot(2,2,3);
```

```
plot(s2);
title('默认阈值去噪后的信号');grid;
%用给定的软阈值进行去噪处理
cd1soft=wthresh(cd1,'s',1.465);
cd2soft=wthresh(cd2,'s',1.823);
cd3soft=wthresh(cd3,'s',2.768);
c2=[ca3 cd3soft cd2soft cd1soft];
s3=waverec(c2,1,'db1');
subplot(2,2,4);
plot(s3);
title('给定软阈值去噪后的信号');grid;
```

信号去噪结果如图 5-3 所示。

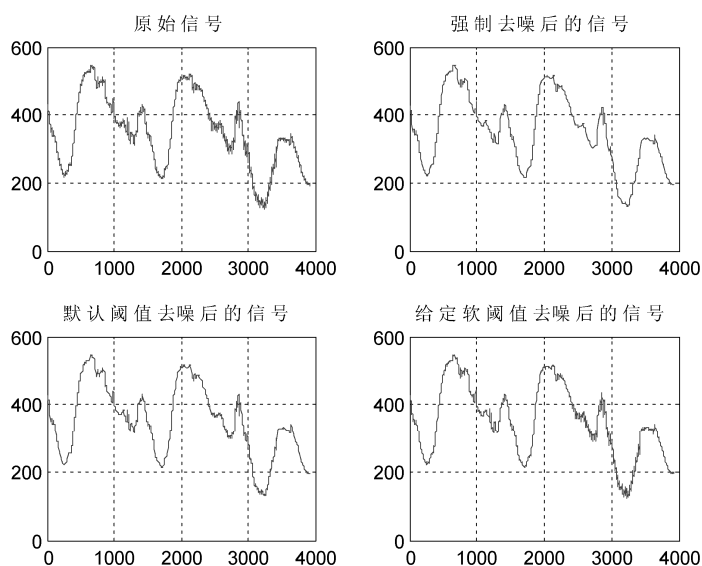


图 5-3 含噪正弦波去噪结果

从图 5-3 得到的结果来看，应用强制去噪处理后的信号较为光滑，但它很有可能丢失了信号中的一些有用成分。默认阈值去噪和给定软阈值去噪这两种处理方法在实际中应用得更为广泛。

【例 5-4】计算高斯白噪声信号  $N(0,1)$  在上述 4 条规则下的阈值。

```
%产生白噪声
y=randn(1,1000);
%生成不同的阈值
thr1=thselect(y,'rigrsure')
thr2=thselect(y,'sqtwolog')
thr3=thselect(y,'heursure')
thr4=thselect(y,'minimaxi')
```



计算结果如下：

```
thr1 =  
    2.7316  
thr2 =  
    3.7169  
thr3 =  
    3.7169  
thr4 =  
    2.2163
```

在本例程中，信号  $y$  是一个标准的高斯白噪声，所以每种方法都能粗略地将所有系数剔除。从计算的结果来看，对于 Stein 的无偏似然估计（SURE）和极大极小（minimaxi）原理的阈值选择规则，仅保存了约 3% 的系数；而其两种阈值选择规则将所有的系数都变成了零。

同样地，对噪声进行小波分解时也会产生高频系数，故一个信号的高频系数向量是有用信号和噪声信号的高频系数的叠加。由于 SURE 和 minimaxi 阈值选取规则较为保守（仅将部分系数置为零），因此在信号的高频信息有很少一部分在噪声范围内时，这两种阈值非常有用，可以将弱小的信号提取出来。其他两种阈值选取规则在去除噪声时更为有效，但也可能将有用信号的高频部分当作噪声信号去除掉。

在实际的工程应用中，大多数信号可能包含许多尖峰或突变，而且噪声信号也并不是平稳的白噪声。对这种信号进行去噪处理时，传统的傅里叶变换完全是在频域中对信号进行分析，不能给出信号在某个时间点上的变化情况，因此分辨不出信号在时间轴上的任何一个突变。但是小波分析能同时在时频域内对信号进行分析，所以它能有效地区分信号中的突变部分和噪声，从而实现非平稳信号的去噪。

下面通过一个实例来考查小波分析对非平稳信号的去噪。

**【例 5-5】**利用小波分析对一个含噪的矩形波信号进行去噪处理。

```
%设置信噪比和随机种子值  
snr=4;  
init=2055615866;  
%产生原始信号 sref 和被高斯白噪声污染的信号 s  
[sref,s]=wnoise(1,11,snr,init);  
  
%用 sym8 小波对信号 s 进行三层分解并对细节系数  
%选用 sure 阈值模式和尺度噪声  
xd=wden(s,'heursure','s','one',3,'sym8');  
  
%显示信号波形  
subplot(3,1,1);  
plot(sref);  
xlabel('样本序号 n');  
ylabel('幅值 A');  
title('参考信号');  
subplot(3,1,2);
```



```

plot(s);
title('含噪信号');
xlabel('样本序号 n');
ylabel('幅值 A');
subplot(3,1,3);
plot(xd);
title('去噪信号');
xlabel('样本序号 n');
ylabel('幅值 A');

```

计算结果如图 5-4 所示。

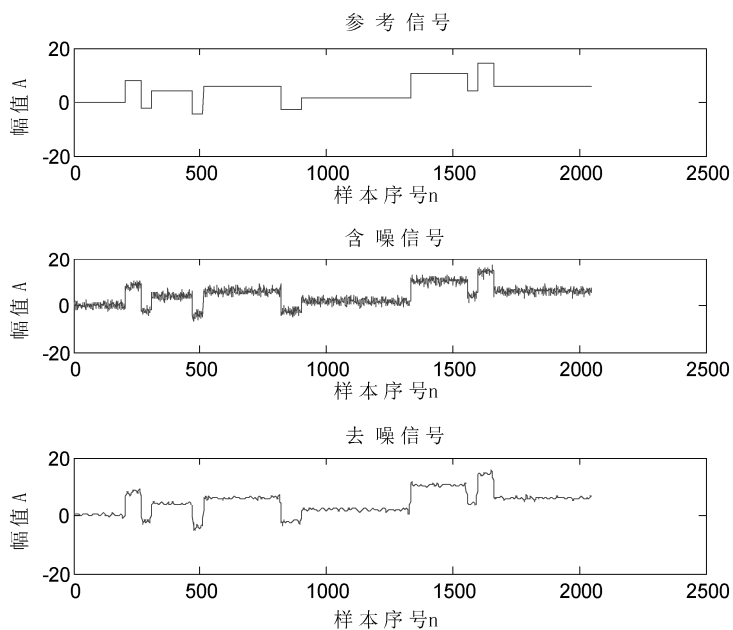


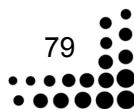
图 5-4 含噪矩形波的去噪结果

【例 5-6】利用小波分析对含噪正弦波进行去噪。

```

%生成正弦信号
N=1000;
t=1:N;
x=sin(0.03*t);
%加噪声
load noissin;
ns=noissin;
%显示波形
subplot(3,1,1);
plot(t,x);
title('原始信号');
subplot(3,1,2);

```







```
plot(ns);  
title('含噪信号');  
%小波去噪  
xd=wden(ns,'minimaxi','s','one',5,'db3');  
subplot(3,1,3);  
plot(xd);  
title('去噪信号');
```

信号去噪结果如图 5-5 所示。

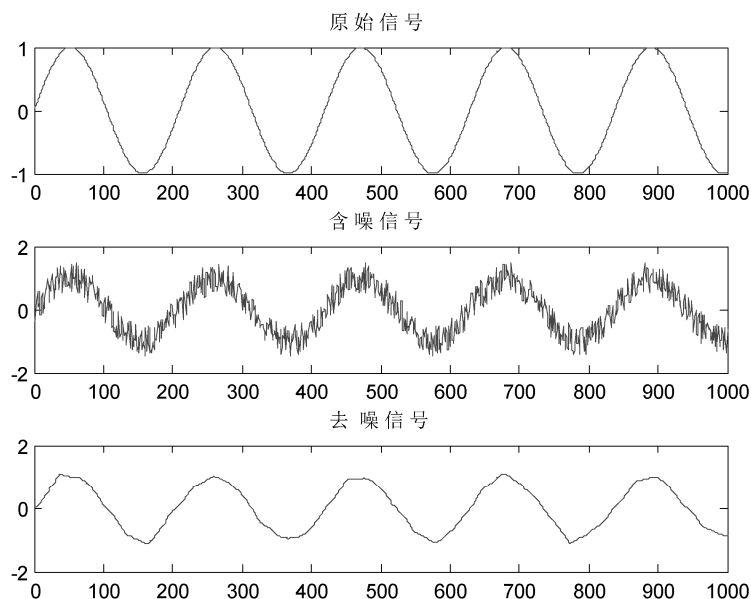


图 5-5 含噪正弦波去噪结果

从图 5-5 中可以看出，去噪后的信号大体上恢复了原始信号的形状，并明显地除去了噪声所引起的干扰。但是，恢复后的信号和原始信号相比，有明显的改变。这主要是在进行去噪处理的过程中所用的分析小波和细节系数阈值不恰当所致。下面我们再通过一个例程对此进行说明。

在 MATLAB 的小波工具箱中，设置软或硬阈值的函数为 `wthresh.m`。该函数根据参数 `sorh` 的值计算分解系数的软阈值或硬阈值。其中，硬阈值对应于最简单的处理方法，而软阈值具有很好的数学特性，并且所得到的理论结果是可用的。

【例 5-7】生成不同阈值下的信号。

```
%生成线性信号  
y=linspace(-1,1,100);  
%设置 T 阈值  
thr=0.4;  
%计算软、硬阈值  
ythard=wthresh(y,'h',thr);  
ytsoft=wthresh(y,'s',thr);
```

```
%显示不同阈值后的信号
subplot(1,3,1);
plot(y);
title('原始信号');grid;
subplot(1,3,2);
plot(ythard);
title('硬阈值信号');grid;
subplot(1,3,3);
plot(ytsoft);
title('软阈值信号');grid;;
```

去噪结果如图 5-6 所示。

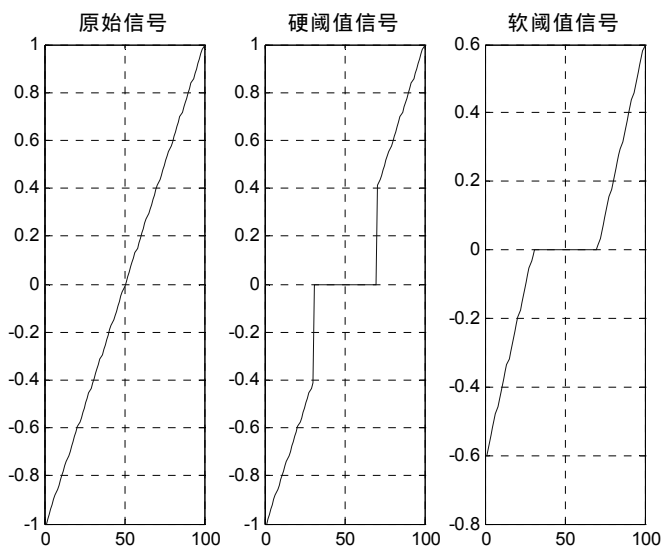


图 5-6 不同阈值下的信号

说明：令  $t$  表示阈值，则硬阈值信号  $s$  的形式为  $s = \begin{cases} x & |x| > t \\ 0 & |x| \leq t \end{cases}$ ，软阈值信号  $s$  的形式为  $s = \begin{cases} \text{sig}[n(x)(|x| - t)] & |x| > t \\ 0 & |x| \leq t \end{cases}$ 。

根据基本的噪声模型，阈值的选取有 4 个规则，分别对应于函数 `thselect` 中输入参数 `tpr` 的一个选项。

### 5.3 信号分析与检测

小波变换作为信号处理的一种手段，逐渐被越来越多领域的理论工作者和工程技术人员所重视和应用，并在许多应用中取得了显著的效果。



本节主要介绍如何利用小波分析来分析信号。这里仅对某类小波分析结果做出了一定的分析，有兴趣的读者可以针对上述各类信号使用的小波分析方法，观察所得到的结果，以加深小波分析用于信号处理的基本步骤和过程。

含噪的三角波与正弦波的组合，其表达式为

$$s(t) = \begin{cases} \frac{t-1}{500} + \sin 0.3t + b(t) & 1 \leq t \leq 500 \\ \frac{1000-t}{500} + \sin 0.3t + b(t) & 501 \leq t \leq 1000 \end{cases}$$

【例 5-8】应用 db5 小波对上式正弦波组合信号进行 7 层分解。

```
%生成正弦信号
N=1000;
t=1:N;
sig1=sin(0.3*t);
%生成三角波信号
sig2(1:500)=(1:500)-1)/500;
sig2(501:N)=(1000-(501:1000))/500;
figure(1);
subplot(2,1,1);
plot(t,sig1,'LineWidth',2);
xlabel('样本序号 n');
ylabel('幅值 A');
subplot(2,1,2);
plot(t,sig2,'LineWidth',2);
xlabel('样本序号 n');
ylabel('幅值 A');
%叠加信号
x=sig1+sig2+randn(1,N);
figure(2)
plot(t,x,'LineWidth',2);
xlabel('样本序号 n');
ylabel('幅值 A');
%一维小波分解
[c,l] = wavedec(x,7,'db5');

%重构第 1 ~ 7 层逼近系数.
a7 = wrcoef('a',c,l,'db5',7);
a6 = wrcoef('a',c,l,'db5',6);
a5 = wrcoef('a',c,l,'db5',5);
a4 = wrcoef('a',c,l,'db5',4);
a3 = wrcoef('a',c,l,'db5',3);
a2 = wrcoef('a',c,l,'db5',2);
```



```
a1 = wrcoef('a',c,l,'db5',1);
```

```
%显示逼近系数
```

```
figure(3)
```

```
subplot(7,1,1);
```

```
plot(a7,'LineWidth',2);
```

```
ylabel('a7');
```

```
subplot(7,1,2);
```

```
plot(a6,'LineWidth',2);
```

```
ylabel('a6');
```

```
subplot(7,1,3);
```

```
plot(a5,'LineWidth',2);
```

```
ylabel('a5');
```

```
subplot(7,1,4);
```

```
plot(a4,'LineWidth',2);
```

```
ylabel('a4');
```

```
subplot(7,1,5);
```

```
plot(a3,'LineWidth',2);
```

```
ylabel('a3');
```

```
subplot(7,1,6);
```

```
plot(a2,'LineWidth',2);
```

```
ylabel('a2');
```

```
subplot(7,1,7);
```

```
plot(a1,'LineWidth',2);
```

```
ylabel('a1');
```

```
xlabel('样本序号 n');
```

```
%重构第 1 ~ 7 层细节系数
```

```
d7 = wrcoef('d',c,l,'db5',7);
```

```
d6 = wrcoef('d',c,l,'db5',6);
```

```
d5 = wrcoef('d',c,l,'db5',5);
```

```
d4 = wrcoef('d',c,l,'db5',4);
```

```
d3 = wrcoef('d',c,l,'db5',3);
```

```
d2 = wrcoef('d',c,l,'db5',2);
```

```
d1 = wrcoef('d',c,l,'db5',1);
```

```
%显示细节系数
```

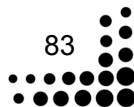
```
figure(4)
```

```
subplot(7,1,1);
```

```
plot(d7,'LineWidth',2);
```

```
ylabel('d7');
```

```
subplot(7,1,2);
```





```
plot(d6,'LineWidth',2);  
ylabel('d6');  
subplot(7,1,3);  
plot(d5,'LineWidth',2);  
ylabel('d5');  
subplot(7,1,4);  
plot(d4,'LineWidth',2);  
ylabel('d4');  
subplot(7,1,5);  
plot(d3,'LineWidth',2);  
ylabel('d3');  
subplot(7,1,6);  
plot(d2,'LineWidth',2);  
ylabel('d2');  
subplot(7,1,7);  
plot(d1,'LineWidth',2);  
ylabel('d1');  
xlabel('样本序号 n');
```

含噪的三角波与正弦波混合信号波形如图 5-7 所示。利用 db5 小波对混合信号进行 7 层分解，得到的逼近信号如图 5-8 所示。细节信号如图 5-9 所示，可以看出，在图 5-9 中，细节信号 d1 和 d2 是与噪声相关的，而 d3（特别是 d4）是与正弦信号相关的，逼近信号 d7 是一个三角波。

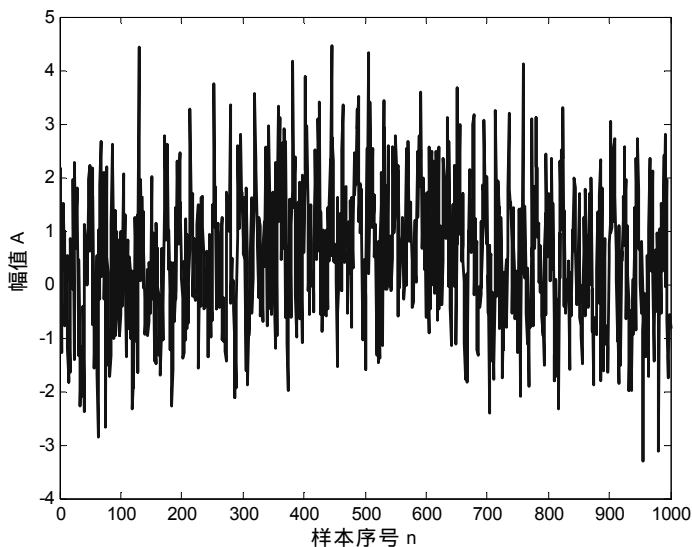


图 5-7 含噪的三角波与正弦波混合信号波形

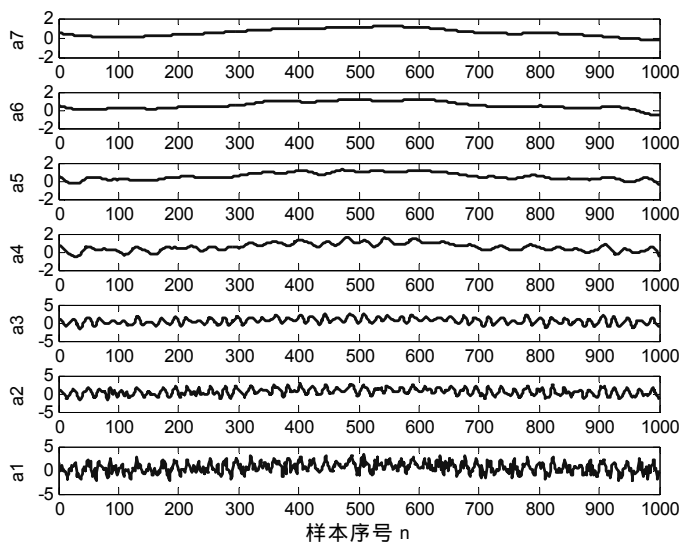


图 5-8 小波分解后各层逼近信号

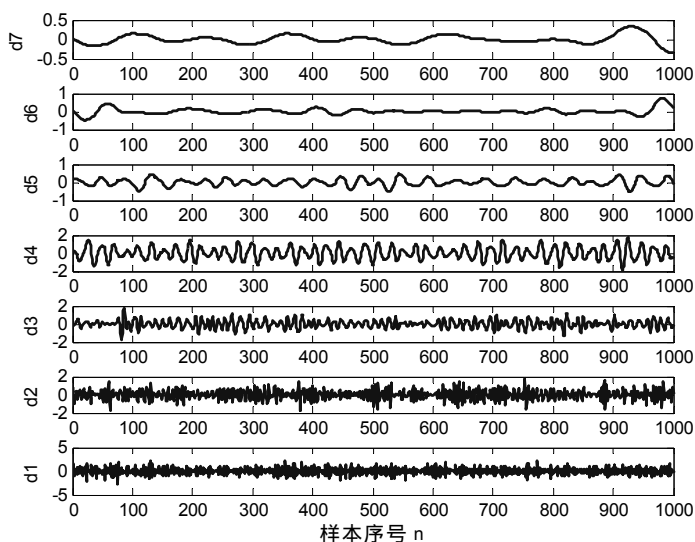
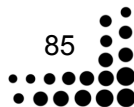


图 5-9 小波分解后各层细节信号

【例 5-9】利用小波分析检测信号中的奇异点并消除。

```
%装载采集的信号 leleccum.mat
load leleccum;
%将信号中第 1160~1235 个采样点赋给 s
index=1160:1235;
s=leleccum(index);
%画出原始信号
figure(1);
plot(index,s);
```





```
xlabel('样本序号 n');
ylabel('幅值 A');

%用 db3 小波进行 5 层分解
[c,l]=wavedec(s,5,'db3');
%重构第 1~5 层逼近信号
a5 = wrcoef('a',c,l,'db3',5);
a4 = wrcoef('a',c,l,'db3',4);
a3 = wrcoef('a',c,l,'db3',3);
a2 = wrcoef('a',c,l,'db3',2);
a1 = wrcoef('a',c,l,'db3',1);
%重构第 1~5 层细节信号
d5 = wrcoef('d',c,l,'db3',5);
d4 = wrcoef('d',c,l,'db3',4);
d3 = wrcoef('d',c,l,'db3',3);
d2 = wrcoef('d',c,l,'db3',2);
d1 = wrcoef('d',c,l,'db3',1);
%显示细节信号
figure(2)
subplot(3,1,1);
plot(index,d3,'LineWidth',2);
ylabel('d3');
subplot(3,1,2);
plot(index,d2,'LineWidth',2);
ylabel('d2');
subplot(3,1,3);
plot(index,d1,'LineWidth',2);
ylabel('d1');
xlabel('样本序号 n');
%消除奇异点
%%设置第 1~3 层细节信号为零
s0=a5+d5+d4;
%画出重构信号
figure(3);
plot(index,s0);
xlabel('样本序号 n');
ylabel('幅值 A');
```

采集的原始信号波形如图 5-10 所示，可以明显看到，在  $t=1193$  和  $t=1215$  两处存在奇异值点。进一步利用 db3 小波对信号进行 5 层分解，得到的第 1~3 层细节信号如图 5-11 所示，发现奇异值点包含在细节信号 d1 和 d2，且与原信号中的奇异点是同步的。为了消除奇异点，重构信号时令细节信号 d1、d2 和 d3 等于零，得到的信号波形如图 5-12 所示。比较图 5-10 和图 5-12 可见，奇异值点已经很不明显了。

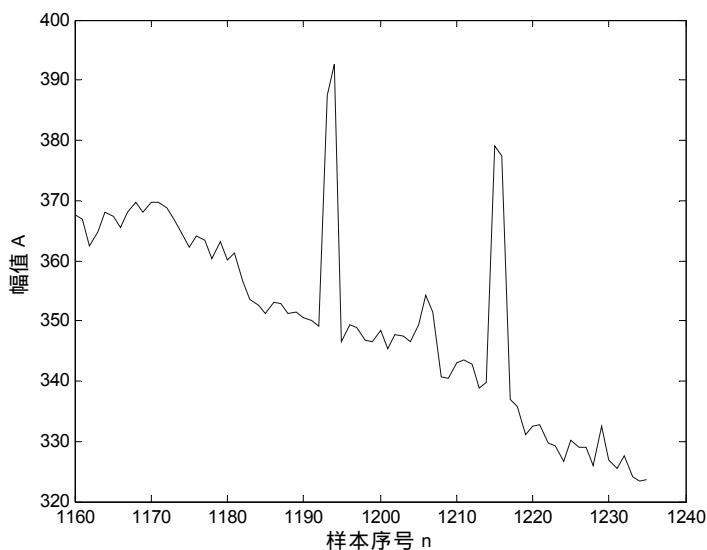


图 5-10 原始信号波形

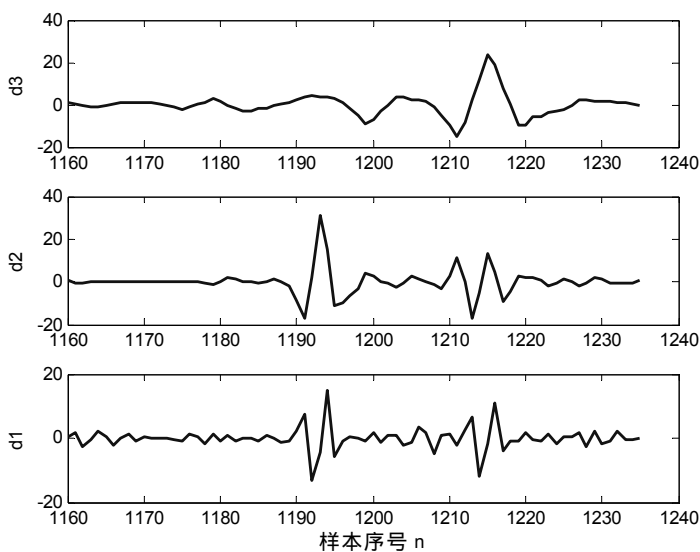


图 5-11 小波分解的细节信号波形

下面的实例测量数据是从一个复杂的设备上采集的电力负载信号，每分钟采集 1 个样本，持续了 5 星期，总共 50400 个数据样本。测量数据受到传感器误差和状态噪声两种噪声的影响。本小节将分析其中的两段数据，其中第一段是上午 12:30 至下午 1:00 采集的样本，由于这段时间处于用电高峰，因此数据很复杂；第二段是下半夜采集的样本，数据比较简单。



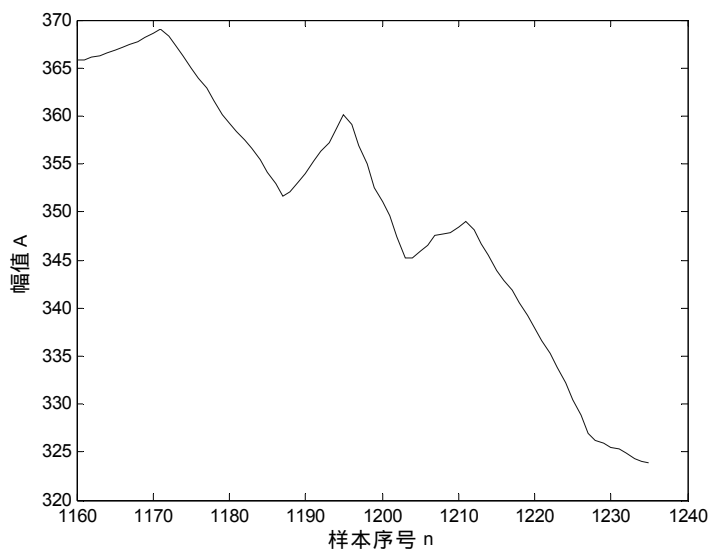


图 5-12 消除奇异点后的波形

**【例 5-10】利用小波分析检测第二段信号的突变点。**

```
%装载采集的信号 leleccum.mat
load leleccum;
%将信号中第 3600 ~ 3700 个采样点赋给 s
index=1575:1720;
s=leleccum(index);
%画出原始信号
figure(1);
plot(index,s);
xlabel('样本序号 n');
ylabel('幅值 A');
%用 db3 小波进行 5 层分解
[c,l]=wavedec(s,5,'db3');
%重构第 1 ~ 5 层逼近系数.
a5 = wrcoef('a',c,l,'db3',5);
a4 = wrcoef('a',c,l,'db3',4);
a3 = wrcoef('a',c,l,'db3',3);
a2 = wrcoef('a',c,l,'db3',2);
a1 = wrcoef('a',c,l,'db3',1);
%显示逼近系数
figure(2)
subplot(5,2,1);
plot(index,a5,'LineWidth',2);
ylabel('a5');
subplot(5,2,3);
plot(index,a4,'LineWidth',2);
```



```

ylabel('a4');
subplot(5,2,5);
plot(index,a3,'LineWidth',2);
ylabel('a3');
subplot(5,2,7);
plot(index,a2,'LineWidth',2);
ylabel('a2');
subplot(5,2,9);
plot(index,a1,'LineWidth',2);
ylabel('a1');
xlabel('样本序号 n');
%重构第 1 ~ 5 层细节系数
d5 = wrcoef('d',c,l,'db3',5);
d4 = wrcoef('d',c,l,'db3',4);
d3 = wrcoef('d',c,l,'db3',3);
d2 = wrcoef('d',c,l,'db3',2);
d1 = wrcoef('d',c,l,'db3',1);
%显示细节系数
subplot(5,2,2);
plot(index,d5,'LineWidth',2);
ylabel('d5');
subplot(5,2,4);
plot(index,d4,'LineWidth',2);
ylabel('d4');
subplot(5,2,6);
plot(index,d3,'LineWidth',2);
ylabel('d3');
subplot(5,2,8);
plot(index,d2,'LineWidth',2);
ylabel('d2');
subplot(5,2,10);
plot(index,d1,'LineWidth',2);
ylabel('d1');
xlabel('样本序号 n');

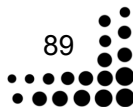
```

第二段电力载波信号如图 5-13 所示，该段数据样本在时间  $t=1600$  和  $t=1625$  两处存在突变点。

利用 db3 小波对其进行 5 层分解，得到的逼近信号和细节如图 5-14 所示，可以看出，由细节信号 d2 可以检测突变点位置  $t=1625$ ，由细节信号 d1 也能隐约看出  $t=1600$  处的突变点。

含噪的多项式信号表达式为

$$s(t)=t^2 - t+1+b_1(t)$$



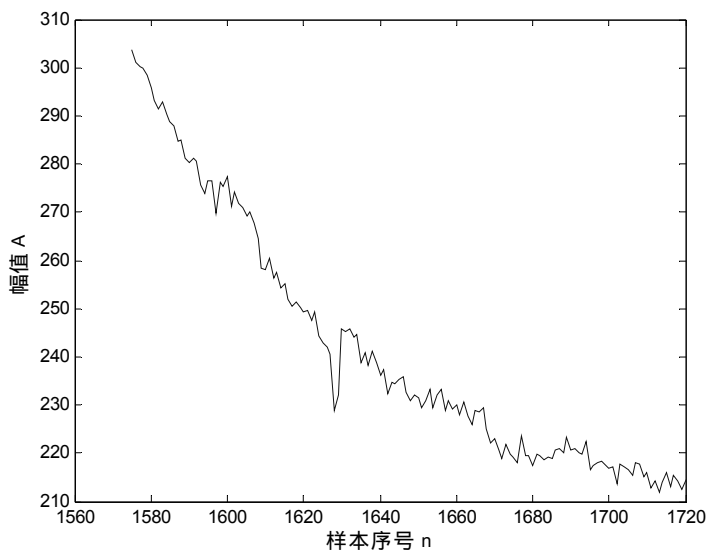


图 5-13 第二段数据波形

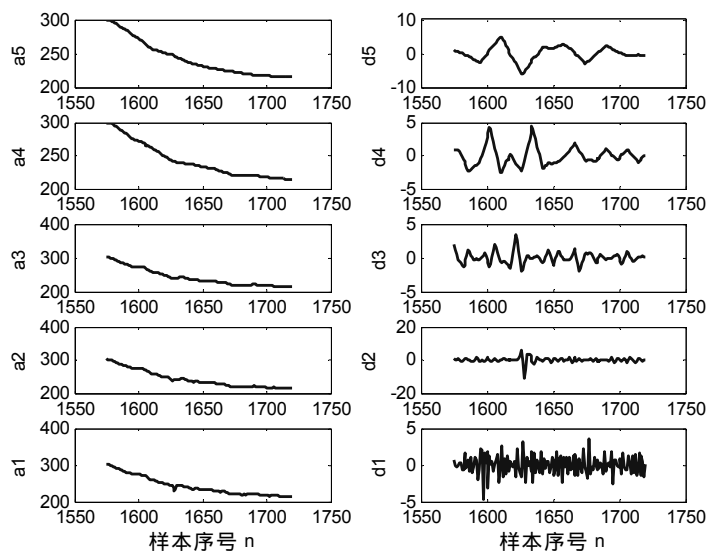


图 5-14 第二段数据小波分析结果

【例 5-11】分别利用 db2 和 db3 小波对上例中含噪的多项式信号进行 4 层分解。

```
%生成含噪的多项式信号
```

```
N=800;
```

```
t=1:N;
```

```
sig=t.^2-t+1;
```

```
x=sig+randn(1,N);
```

```
%一维小波分解
```

```
[c,l]=wavedec(x,4,'db3');
```

```
%[c,l]=wavedec(x,4,'db2');
```



```
%重构第 1 ~ 4 层逼近系数.
```

```
a4 = wrcoef('a',c,l,'db3',4);
```

```
a3 = wrcoef('a',c,l,'db3',3);
```

```
a2 = wrcoef('a',c,l,'db3',2);
```

```
a1 = wrcoef('a',c,l,'db3',1);
```

```
%显示逼近系数
```

```
figure(1)
```

```
subplot(4,1,1);
```

```
plot(a4,'LineWidth',2);
```

```
ylabel('a4');
```

```
subplot(4,1,2);
```

```
plot(a3,'LineWidth',2);
```

```
ylabel('a3');
```

```
subplot(4,1,3);
```

```
plot(a2,'LineWidth',2);
```

```
ylabel('a2');
```

```
subplot(4,1,4);
```

```
plot(a1,'LineWidth',2);
```

```
ylabel('a1');
```

```
xlabel('样本序号 n');
```

```
%重构第 1 ~ 4 层细节系数
```

```
d4 = wrcoef('d',c,l,'db3',4);
```

```
d3 = wrcoef('d',c,l,'db3',3);
```

```
d2 = wrcoef('d',c,l,'db3',2);
```

```
d1 = wrcoef('d',c,l,'db3',1);
```

```
%显示细节系数
```

```
figure(2)
```

```
subplot(4,1,1);
```

```
plot(d4,'LineWidth',2);
```

```
ylabel('d4');
```

```
axis([0 N -100 100]);
```

```
subplot(4,1,2);
```

```
plot(d3,'LineWidth',2);
```

```
ylabel('d3');
```

```
axis([0 N -30 30]);
```

```
subplot(4,1,3);
```

```
plot(d2,'LineWidth',2);
```

```
ylabel('d2');
```

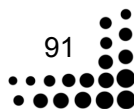
```
axis([0 N -5 5]);
```

```
subplot(4,1,4);
```

```
plot(d1,'LineWidth',2);
```

```
ylabel('d1');
```

```
xlabel('样本序号 n');
```





利用 db2 小波分解后的逼近信号如图 5-15 所示，细节信号如图 5-16 所示。可以看出，这种情况下，随着分解层级的增加，其正则性增加，从而抑制了该多项式信号的零阶和一阶部分，而仅对该信号的二阶部分以及噪声进行了分解，因此在图 5-16 中，除了细节信号 d1 中包含了该含噪信号的不规则性，其余各层细节中的信号周期性（规则性）都是随着层级的增加而增大。

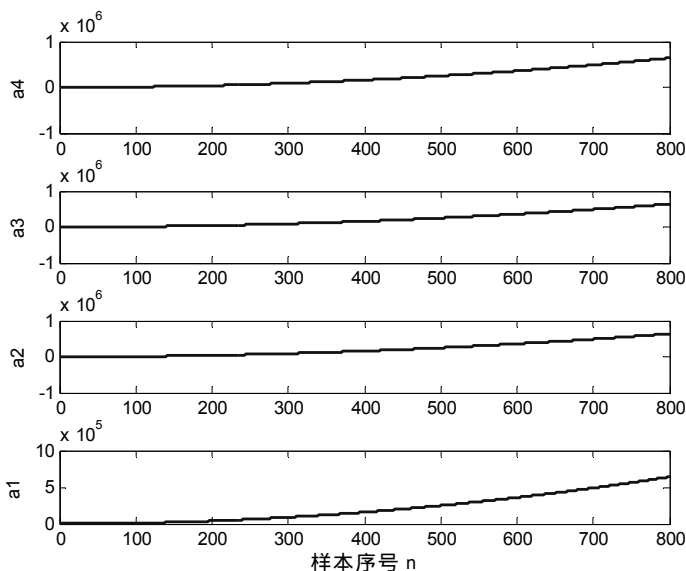


图 5-15 小波分解后各层逼近信号 (db2)

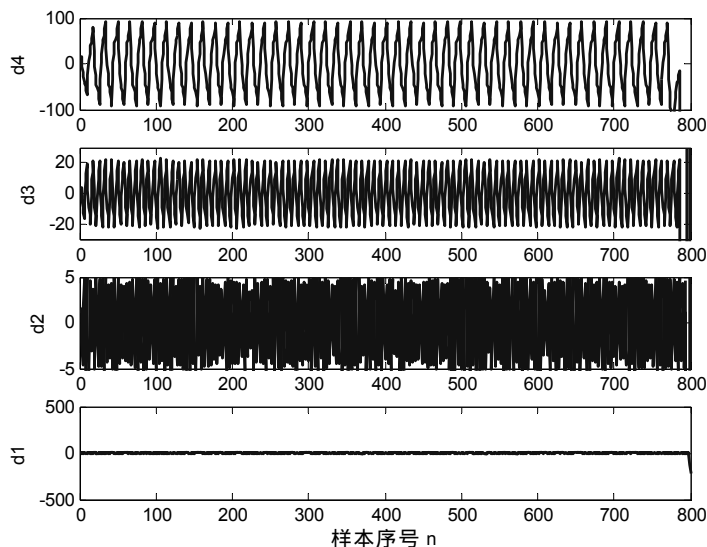


图 5-16 小波分解后各层细节信号 (db2)

利用 db3 小波分解后的逼近信号如图 5-17 所示，细节信号如图 5-18 所示。可以看出，由于 db3 小波的正则性较差，所以它抑制了该信号的多项式部分，而析出了它的噪声部分。

因此，利用小波分析可以较好地对该类信号抑制。

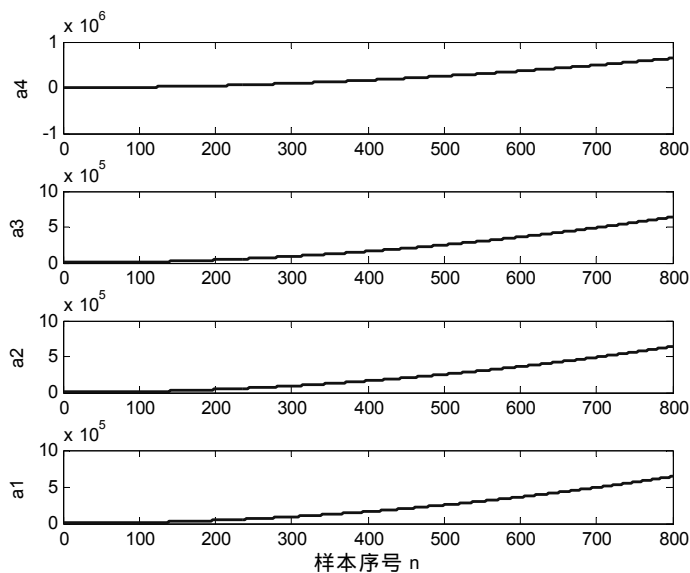


图 5-17 小波分解后各层逼近信号 (db3)

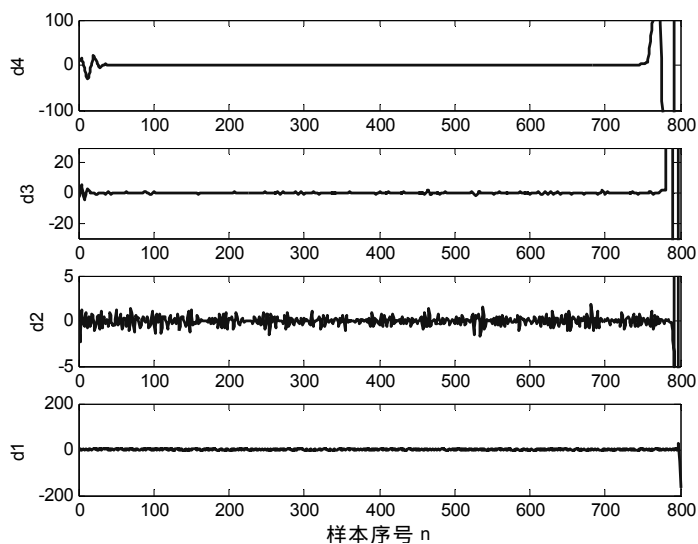


图 5-18 小波分解后各层细节信号 (db3)

## 第6章 基于小波的间断点检测算法分析

时频方法中,现在发展较快的是利用经验模态分解方法(EMD法)求解模态参数的HHT方法,另一种是连续小波模态参数辨识方法。由传感器所检测到的奇异信号往往载有设备运行状态特征的重要信息。判断状态信号的奇异点的出现时刻,并对信号奇异性实现定量描述,在信号处理和故障诊断等领域有重要的意义。

### 6.1 奇异性概念

宇宙射线和太阳黑子爆发、空间核磁爆等对于在太空飞行的卫星和飞船安全构成重大威胁,影响太空飞行器的使用寿命。通过处理采集的空间数据,检测到奇异点,找到太空天气异动的时刻,可以做出科学的判断,及时调整飞行器姿态,从而保护飞行器的安全。寻找变化周期,总结规律,可为进行太空天气预报提供依据。

数学上称无限次可导函数是光滑的或没有奇异性;如果函数在某些有间断或某阶导数不连续,则称函数在此处奇异性,该点就是奇异点。信号的突变点往往包含重要的信息。

一般用Lipschitz指数来刻画信号的奇异性。Lipschitz  $\alpha$  越大,函数越光滑。

设有非负整数  $n (n \leq a \leq n+1)$ , 如果存在着两个常数  $C > 0$  和  $x_0 > 0$  与一个  $n$  阶多项式  $P_n(x)$ , 对于  $x \in (x_0 - \delta, x_0 + \delta)$  使得

$$|f(x_0 + x) - P_n(x)| \leq C|x|^\alpha$$

则称函数  $f$  在点  $x_0$  是 Lipschitz  $\alpha$  的。

(1) 如果存在一个常数  $C$  与一个  $n$  阶的多项式  $P_n$  使得对于任意  $x_0 < x$ , 上式都成立, 则称函数在区间  $(a, b)$  上是一致 Lipschitz  $\alpha$  的。

(2) 函数  $f(x)$  在点  $x_0$  上的 Lipschitz 指数为所有满足上式中  $\alpha$  的上确界。

(3) 如果函数  $f(x)$  在  $x_0$  上的 Lipschitz  $r$  指数  $\alpha$  小于 1, 则称函数在该点是奇异的。

Lipschitz 指数给出了信号  $f(x)$  在  $x_0$  点可导性的精确信息, 某点连续可微的函数  $f(x)$  在该点上的 Lipschitz 指数为 1。如果某点上函数  $f(x)$  的微分有界但不连续, 则  $f(x)$  在该点的 Lipschitz 指数仍然为 1。

可以证明, 当  $f(x)$  在  $x_0$  上的 Lipschitz 指数  $\alpha > 1$  时,  $f(x)$  在  $x_0$  是  $n$  次可导的, 而多项式  $P_n$  是  $f(x)$  在  $x_0$  上的泰勒展开式的前  $n+1$  项。

如果  $n=0$  则  $P_n = f(x_0)$ 。由定义可得知,当  $f(x)$  的 Lipschitz 指数  $\alpha$  满足条件  $n < \alpha < n+1$  时,  $f(x)$  是  $n$  阶可微的,它的第  $n$  阶导数是奇异的,即  $f(x)$  的  $n+1$  阶导数发散。此时,如果  $f(x)$  的  $n$  阶导数有界,则  $f(x)$  的  $n$  阶导数是正则的。如果  $f(x)$  的 Lipschitz 指数为  $\alpha$ ,则每积分一次 Lipschitz 指数增加 1,每微分一次 Lipschitz 指数指数减少 1。

## 6.2 第一类间断点检测

当小波函数可看作某一平滑函数的一阶导数时,信号小波变换模的局部极值点对应于信号的突变点;当小波函数可看作某一平滑函数的二阶导数时,信号小波变换的过零点对应于信号的突变点。因此,采用小波变换模的过零点和局部极值点的方法可以检测信号的突变点。比较来说,用局部极值点的方法进行检测更具优越性。

一般信号奇异性分为两种情况:

(1) 信号在某时刻幅值发生突变,引起信号的不连续。这种类型的突变称为第一种类型的间断点。

(2) 信号外观上很光滑,幅值没有发生突变,但是信号的一阶微分有突变发生且一阶不连续。这种类型的突变称为第二种类型的间断点。

应用小波分析可以检测出信号中突变点的位置、类型以及变化的幅度。

**【例 6-1】**检测信号幅值变化的准确时间或间断点的准确位置。

```
>> clear all;
load freqbrk;                                %载入信号
s=freqbrk;
ls=length(s);
%利用 db4 小波对信号进行 7 层分解
[c,l]=wavedec(s,7,'db4');
subplot(211);plot(s);
title('原始信号');
ylabel('s');
%利用 db4 小波提取第 7 层的近似系数
a7=wrcoef('a',c,l,'db7',7);
subplot(212);plot(a7);
ylabel('a7');
title('用 db 小波分解 7 层的近似系数');      %效果如图 6-1 所示
figure;
for i=1:7
    decmp=wrcoef('d',c,l,'db4',8-i);
    subplot(7,1,i);
    plot(decmp);
    ylabel(['d',num2str(8-i)]);
end
title('用 db4 小波分解 7 层的细节系数');      %效果如图 6-2 所示
```



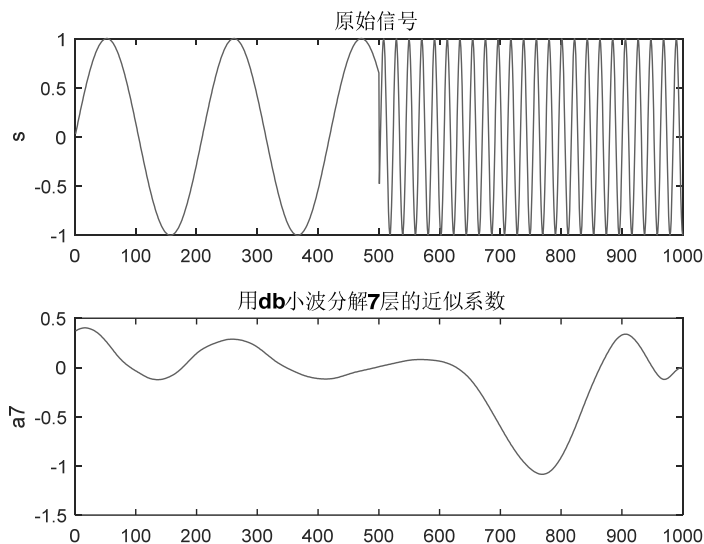


图 6-1 原始信号与近似系数

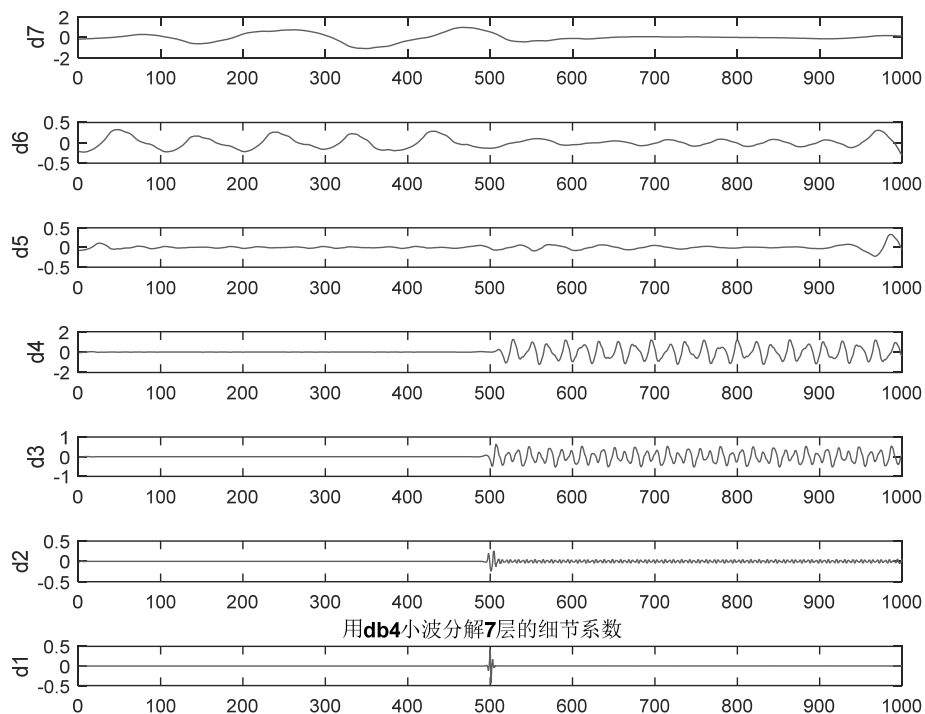


图 6-2 细节系数

注意：本例中信号的不连续是由于低频特征的正弦信号在后半部分突然加入高频特征的正弦信号，分析的目的是将加入高频特征的正弦信号的时间检测出来。

从图 6-2 可看出，信号的不连续点在使用 db4 小波将信号进行 7 层分解来检测第一种类型的间断点，可以非常清楚地观察到信号的不连续点，即高频特征的正弦信号的加入点，这是因为间断点包含了高频信息。

原始信号是由两个独立的满足指数方程的信号在  $t=500$  处连接起来的，因此看上去是光滑的，但它的一阶微分有突变。

采用 db4 小波对信号分解后，在信号的第一层高频系数 d1 中可以明显地看到  $t=500$  的间断点。值得注意的是，在信号奇异点的检测中，选择小波的正则性非常重要，因为这时小波可实现一个长冲激响应滤波器。

**【例 6-2】**用傅里叶变换（FFT）和 db4 小波分别实现对信号在频域内的表面形式，并比较这两种方法的优劣。

```
>> clear all;
load freqbrk;           %载入信号
s=freqbrk;
ls=length(s);
[c,l]=wavedec(s,7,'db4');
figure;                 %效果如图 6-3 所示
subplot(211);plot(s);
title('原始信号');
ylabel('s');
fs=fft(s,1000);        %对信号进行 FFT 变换
fs=abs(fs);
subplot(212);plot(fs);
ylabel('FFT');
grid on;
[c,l]=wavedec(s,3,'db4'); %用 db4 小波进行信号分解
a3=wrcoef('a',c,l,'db4',3); %对分解结构的第 3 层低频进行重构
%对分解的各高频进行重构
figure;                 %效果如图 6-4 所示
for i=1:3
    decmp=wrcoef('d',c,l,'db4',4-i);
    subplot(3,1,i);
    plot(decmp);
    ylabel(['d',num2str(4-i)]);
end
```

从图 6-3 可看出，由于 FFT 将信号变换成纯频域中的信号，使得不具有时间分辨能力，所以对信号在时域中的突变点根本无法检测出来。而 db4 小波分解后的信号则可以很明显地分别出间断点。

对信号进行多尺度分析，在信号出现突变时，其小波变换后的系数具有模量极大值，借此可以检测故障发生的时间。

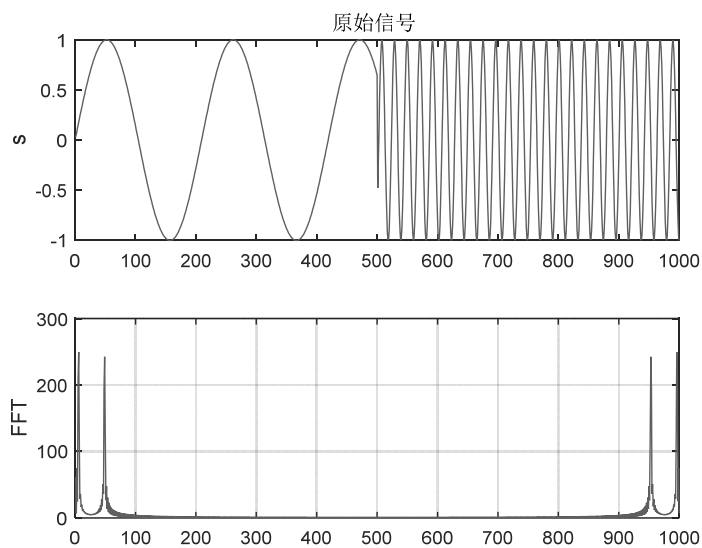


图 6-3 原始信号与 FFT 变换

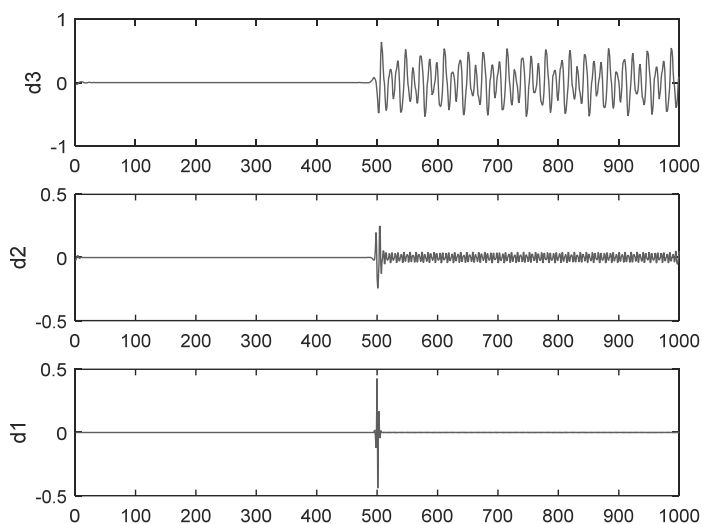


图 6-4 小波变换

【例 6-3】利用小波分解系数检测正弦信号的突变时间起始点。

```
>> clear all;  
t=0:pi/100:2*pi;  
s1=sin(t);  
s2=sin(10*t);  
s3=sin(t);  
%整个信号
```



```

s=[s1 s2 s3];
figure; %效果如图 6-5 所示
subplot(211);plot(s);
title('原始信号');
ylabel('s');

[c,l]=wavedec(s,7,'sym4');
a7=wrcoef('a',c,l,'sym4',7);
subplot(212);plot(a7);
ylabel('a7');
title('用 sym4 小波分解 7 层的近似系数');
figure; %效果如图 6-6 所示
for i=1:7
    decmp=wrcoef('d',c,l,'sym4',8-i);
    subplot(7,1,i);
    plot(decmp);
    ylabel(['d',num2str(8-i)]);
end
title('用 sym4 小波分解 7 层的细节系数');

```

从图 6-6 中的小波分解层系数可明显看出， $t=200 \sim 400$  时，系统工作出现了异常情况。

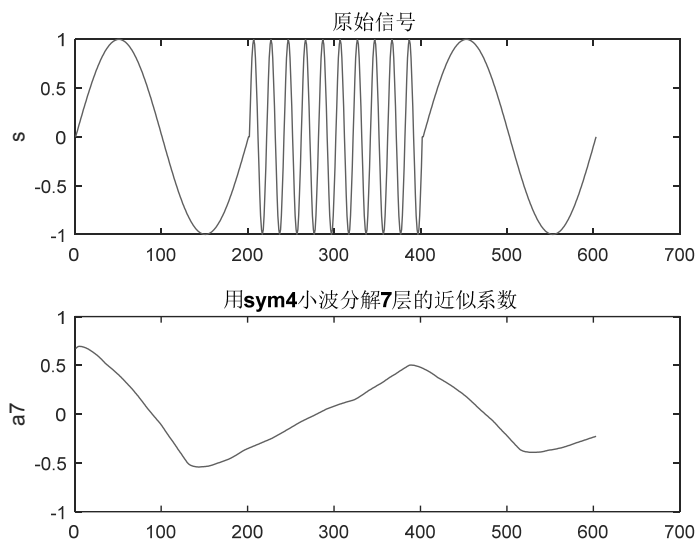
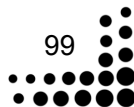


图 6-5 原始信号与近似系数



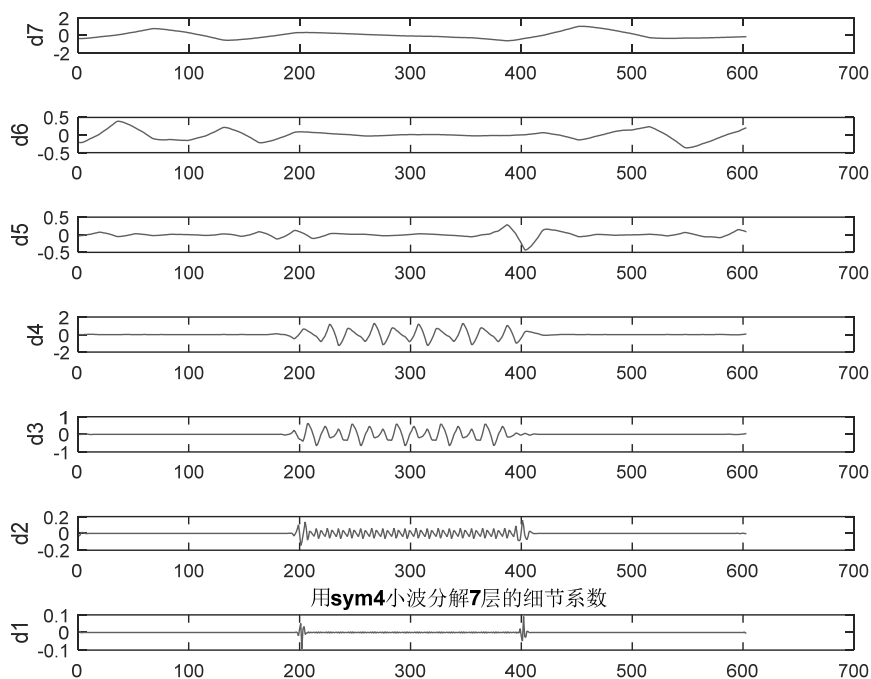


图 6-6 细节系数

## 6.3 第二类间断点检测

6.2 节讨论了信号本身的间断点问题，即第一类间断点。在实际应用中，还有一类重要的间断问题，就是导数的间断，即第二类间断点。这种信号的本身可能是连续的，但信号的导数代表了信号的另一重要信息。比如，对位移信号，一阶导数可能表明了速度信息，二阶导数可能表明了受力信息等。

【例 6-4】对于一给定的线性信号，利用小波分析来检测信号的第二类间断点。

```
>> clear all;
load nearbrk;           %载入信号
s=nearbrk;
figure;                 %效果如图 6-7 所示
subplot(211);plot(s);
title('原始信号');
ylabel('s');
[c,l]=wavedec(s,7,'sym4');
a7=wrcoef('a',c,l,'sym4',7);
subplot(212);plot(a7);
ylabel('a7');
title('用 sym4 小波分解 7 层的近似系数');
figure;                 %效果如图 6-8 所示
```



```

for i=1:7
    decmp=wrcoef('d',c,l,'sym4',8-i);
    subplot(7,1,i);plot(decmp);
    ylabel(['d',num2str(8-i)]);
end

```

title('用sym4小波分解7层的细节系数');

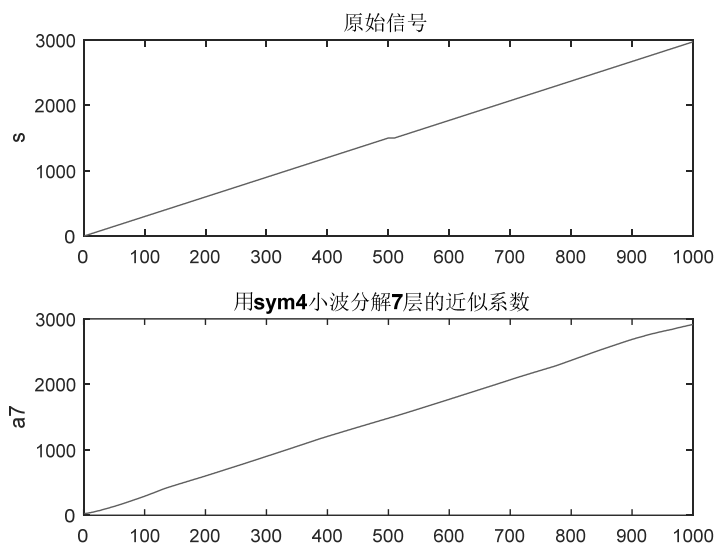


图 6-7 原始信号与近似系数

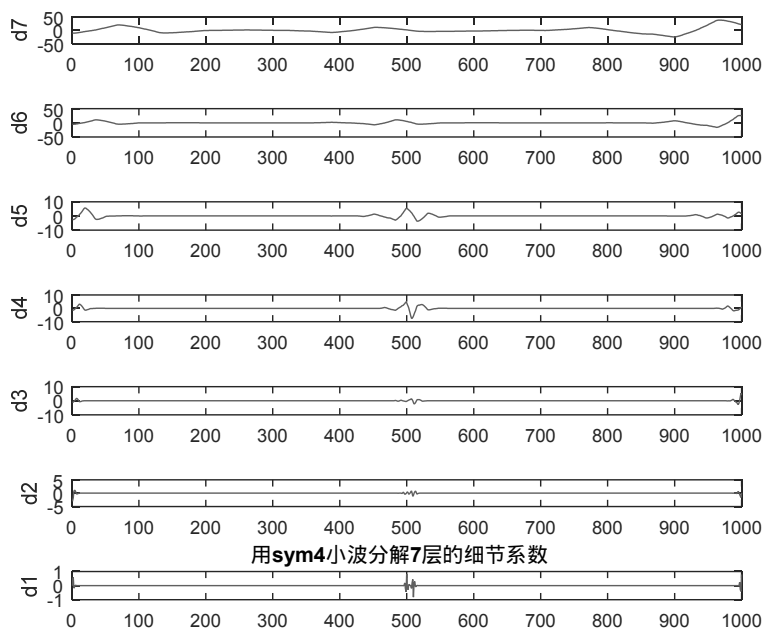
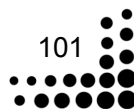


图 6-8 细节系数





本例中，原始信号是一条光滑的直线，但是它一阶微分是突变的。利用 sym4 小波对信号进行 7 层分解后，便将该信号的第二类间断点（ $t=500$ ）检测出来。

另外，所选择的小波基非常重要，本例使用的是正则性小波。读者也可以选择一个不具有正则性的小波进行分析，看是否可以检测出来。同时，也可以选择不同的分解层数，如果分解为 3 层，看是否可以检测出来。

【例 6-5】对一个给定信号，外观上是光滑的，由两个指数方程接合而成，利用小波分析检测其突变的位置。

```
>> clear all;
t=0:0.002:4;
s1=exp(t);
s2=exp(3*t);
s=[s1,s2];
figure(1);           %效果如图 6-9 所示
subplot(211);plot(s);
title('原始信号');
ylabel('s');
%计算信号一阶导数
ds=diff(s);
ylabel('信号导数');
[c,l]=wavedec(s,4,'db4');
a4=wrcoef('a',c,l,'db4',4);
subplot(212);plot(ds);
ylabel('ds');
title('信号导数');
figure(2);           %效果如图 6-10 所示
for i=1:4
    decmp=wrcoef('d',c,l,'db4',5-i);
    subplot(4,1,i);
    plot(decmp);
    ylabel(['d',num2str(5-i)]);
end
```

从图 6-9 中可看出，信号的一阶导数在  $t=2000$  时发生了突变，而原始信号却是光滑的。但经过小波分解后，从图 6-10 中的小波分解层系数可明显看出， $t=2000$  时，信号出现了异常情况。

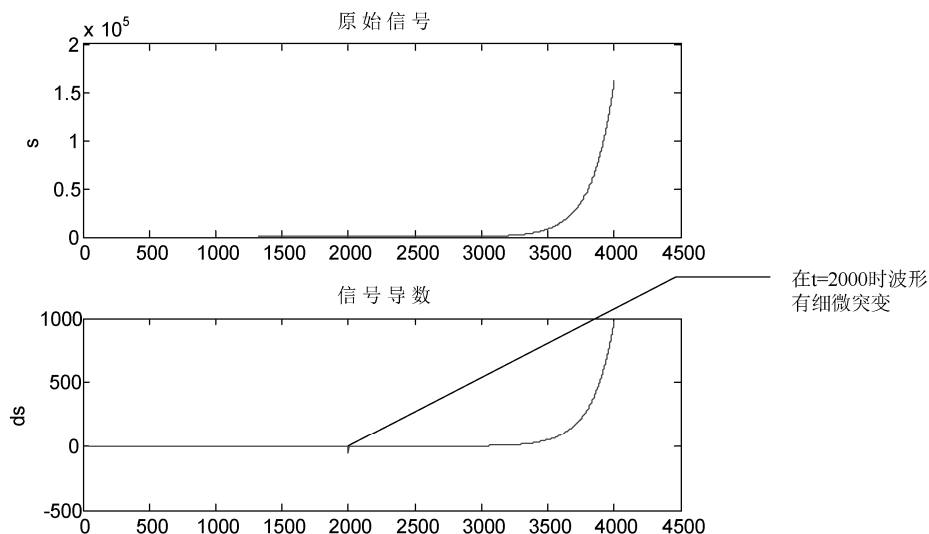


图 6-9 原始信号与对应的导数

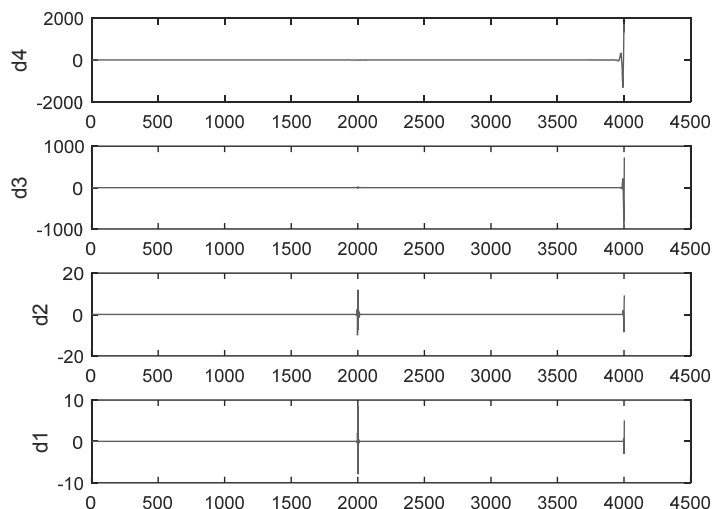


图 6-10 小波分解层数的系数

## 6.4 自相似检测

在连续小波变换中，理论上可以求得在信号上任意小尺度的小波分解系数，因此连续小波变换可以用来研究信号的自相似或分形的性质。

在信号处理的意义上，自相似性指的是信号的某部分在较高的分辨率下采样的结果与信号整体在较低分辨率下的采样结果有一定的相似性。这很符合连续小波变换对任意对尺度进行分解的特点，通过连续小波变换给出的相空间系数，很容易通过系数的分布找到信号的自相似特性。





从小波分解的角度来解释这个问题需要引入相似程度的概念,从直观意义上说,小波分解就是计算一系列信号和小波函数之间的相似系数,相似程度越高则小波系数的绝对值越大,反之则越小。

【例 6-6】利用小波分析检测信号的自相似性。

```
>> clear all;
load scddvbrk;
s=scddvbrk;
figure;
subplot(211);plot(s);
title('原始信号');
ylabel('s');
%计算小波分解自相似指数
subplot(212);f=cwt(s,[1:2:128],'db4','plot');
title('小波分解自相似指数');
xlabel('时间');
ylabel('尺度变换');
```

运行程序,效果如图 6-11 所示。

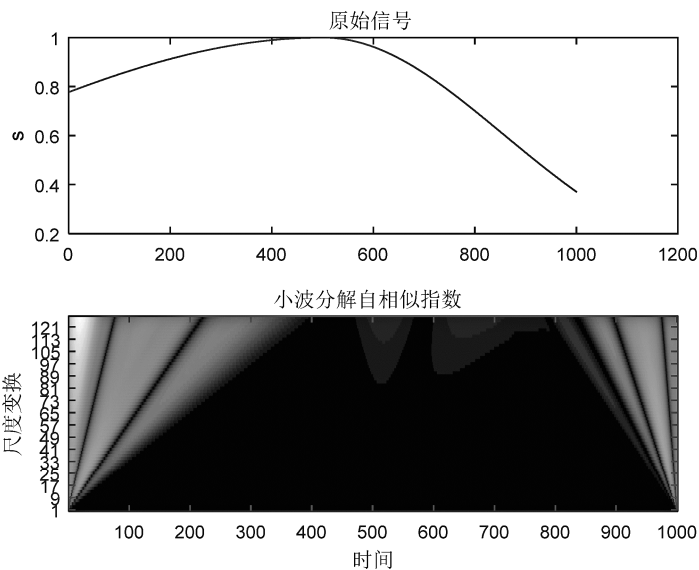


图 6-11 信号的自相似性检测

目前,许多人正在做这项工作,其结果表明,采用小波分解可以很好地研究信号或图像的分形特征。若分形特征开始时随着时间的发展而变换,然后又不变时,则这种信号被称为多分形。小波分析工具非常适合于分形的实际研究和分形的生成。

## 6.5 信号的识别

一些受噪声影响的信号，其发展趋势很难分辨。还有一类信号，经常需要提取其特征来识别其突变程度。

由于噪声的干扰，对于实用信号的发展趋势在时域中难以看出，但是，通过小波分解可以去除那些干扰信号。

【例 6-7】利用小波分析信号的发展趋势。

```
>> clear all;
t=0:pi/200:1;
s1=sin(t);
n=length(s1);
s2=randn(1,n);
s=s1+s2;
figure;          %效果如图 6-12 所示
plot(s);
title('原始信号');
ylabel('s');
[c,l]=wavedec(s,7,'db4');
figure;          %效果如图 6-13 所示
for i=1:7
    decmp=wrcoef('a',c,l,'db4',8-i);
    subplot(7,1,i);
    plot(decmp);
    ylabel(['a',num2str(8-i)]);
end
```

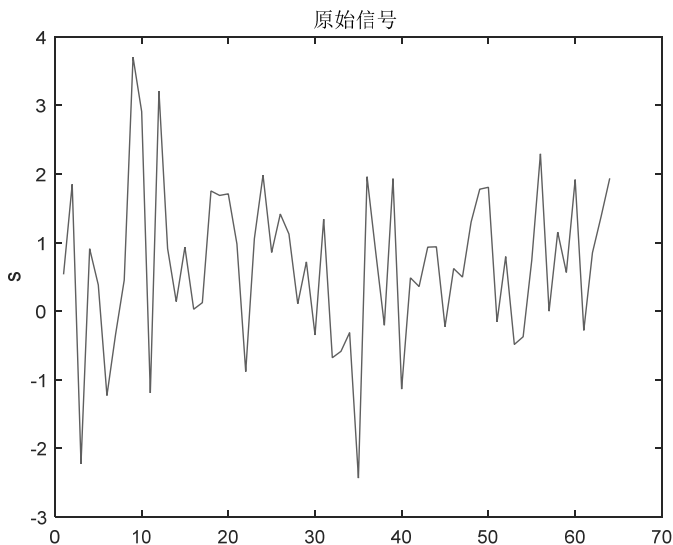


图 6-12 原始信号

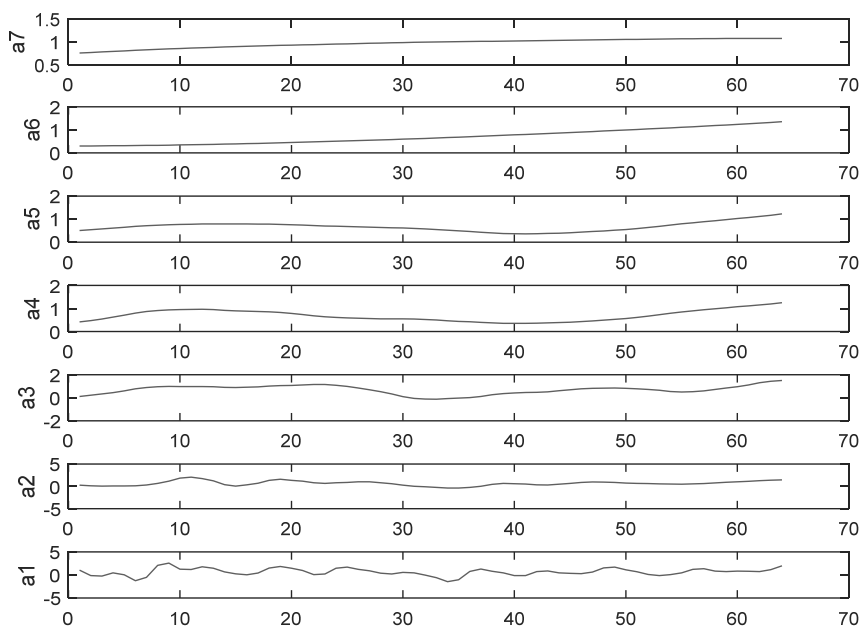


图 6-13 近似分解

从原始信号可看出,由于噪声的污染,信号的发展趋势是不可见的。而在本例中利用 db4 小波分解到第 7 层,信号的发展趋势随着近似变换变得越来越清晰。信号中的低频部分 ( $a_7$ ) 代表着信号的发展趋势。在小波分析中,则对应着最大尺度小波变换的低频系数。因而,随着尺度的增加,时间分辨率降低,信号的发展趋势会表现得更加明显。此外,还可以在频率中理解它的含义,即尺度分解中的低频部分随着层次的增加,其含的高频信息会随之减小。当分解到下一层时,就有一些更高频率信息被滤掉,而剩下的就是信号的发展趋势。

由此可看出,在展示信号的发展趋势下,小波分析是有用的。这种分析所具有的另一个目的是将隐藏在噪声或其他高频信号中的信号显示出来。这里要强调的是,所有识别的信号本身不具有大的突变。这是因为信号的发展趋势是由信号的低频部分所表征的,如果在信号本身中包含有很大突变,那么多尺度小波变换的低频部分中,显示出来的信号会和原始信号有很大差别,因为这种变换将信号本身的突变当作高频信息给滤掉了。

在傅里叶分析中,如果一个信号是由几个不同频率的正弦信号组成,则变换可以很有效地分辨这些不同频率的正弦信号。

**【例 6-8】**利用小波分析将给定信号的各频率成分分开。

```
>> clear all;  
t=0:pi/100:4*pi;  
s1=sin(t);  
s2=sin(4*t);  
s3=sin(45*t);  
s=s1+s2+s3;  
figure; %效果如图 6-14 所示  
plot(s);
```



```

title('原始信号');
ylabel('s');
figure;           %效果如图 6-15 所示
[c,l]=wavedec(s,6,'db4');
%对分解结构中各低频部分进行重构
for i=1:6
    decmp=wrcoef('a',c,l,'db4',7-i);
    subplot(6,2,2*i-1);
    plot(decmp);
    ylabel(['a',num2str(7-i)]);
end
%对分解结构中各高频部分进行重构
for i=1:6
    decmp=wrcoef('d',c,l,'db4',7-i);
    subplot(6,2,2*i);
    plot(decmp);
    ylabel(['d',num2str(7-i)]);
end
%绘制 d1 放大波形图
figure;           %效果如图 6-16 所示
d1=wrcoef('d',c,l,'db4',1);
plot(d1(1:100));

```

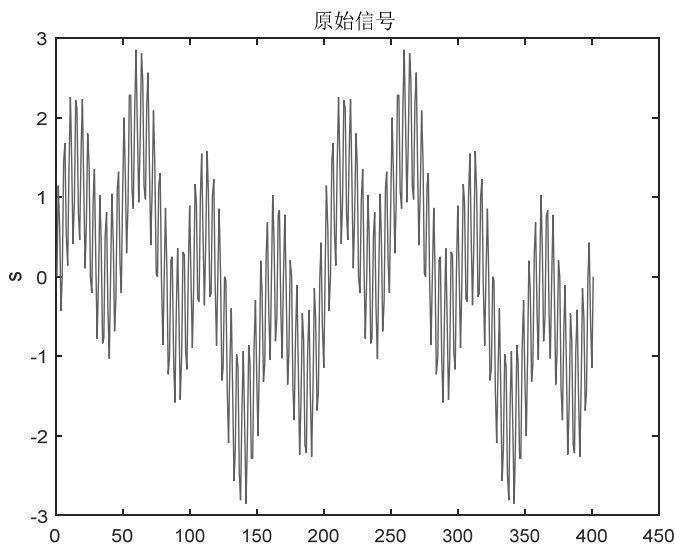
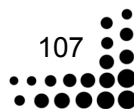


图 6-14 原始信号



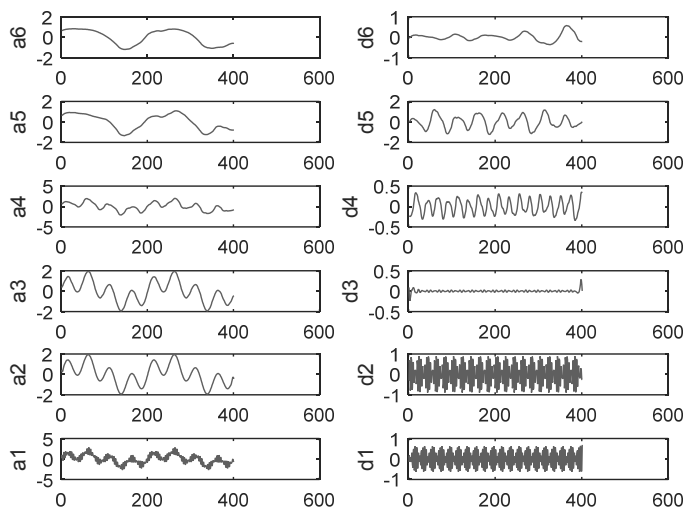


图 6-15 低、高频分解系数

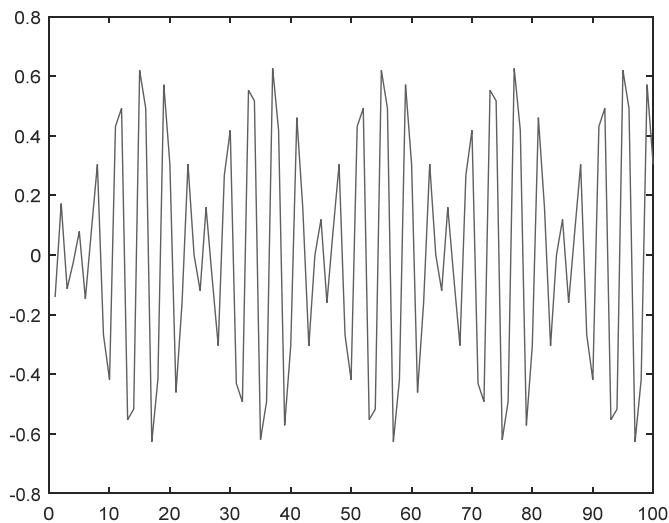


图 6-16 d1 放大效果图

总之，能够用小波分析将合成信号中的单纯正弦信号的频率提取出来。因为在小波分解下，不同的尺度具有不同的时间和频率分辨率，因而能够将信号的不同频率成分分开。

## 第7章 图像的小波分解算法 与实现

小波变换通过多分辨分析过程将一幅图像分解成近似和细节部分，细节对应的是小尺度的瞬间，它在本尺度内很稳定。因此将细节存储起来，对近似部分在下一个尺度上进行分解时重复该过程即可。近似与细节在正交镜像滤波器算法中分别对应于高通和低通滤波器，这种变换通过尺度去掉相关性，在图像压缩中被证明是有效的。由于小波变换后高频部分小波系数的绝对值较小，而低频部分小波系数的绝对值较大，这样，在图像编码处理中，可以对高频部分的大多数系数分配较少的比特数，以达到压缩的目的。

小波分析是近 20 多年发展起来的新兴学科，作为一种快速高效、高精度的近似方法，它是傅里叶分析的一个突破性发展，给许多相关学科的研究领域带来了新的思想，为工程应用提供了一种新的分析工具。小波分析是目前国际上公认的图像/信号信息获取与处理领域的高新技术，是多学科共同关注的热点，是图像/信号处理的前沿课题。

近年来，小波分析理论受到众多学科的共同关注。在最近推出的静态图像压缩国际标准——JPEG2000 中，小波变换（DWT）已经正式取代离散余弦变换（DCT）而成为标准的变换编码方法。本章在上述研究的基础上，提出了一种选择最优的小波基用于图像压缩的评价方法算法，并以仿真实验进行了实例算法研究。

### 7.1 图像的小波分解算法

分解算法实际上就是将一幅图像通过二维小波变换分解成一系列、方向和空间局部变化的子带。一幅图像经小波分解后，可得一系列不同分辨率的子图像，这是对图像进行各种处理的基础，尤其是在图像压缩方面。

设  $\{V_j\}_{j \in \mathbb{Z}}$  是一个二维可分离的多分辨分析， $V_j = V_j^1 \otimes V_j^1$ ，其中  $\{V_j^1\}_{j \in \mathbb{Z}}$  是  $L^2(\mathbb{R})$  上的一个多分辨分析，其尺度函数为  $\phi$ ，小波函数为  $\psi$ ，那么有相应于二维的可分离的尺度函数  $\phi(x, y)$  和 3 个可分离的方向敏感小波函数： $\psi^H(x, y)$ 、 $\psi^V(x, y)$ 、 $\psi^D(x, y)$ ，即

$$\phi(x, y) = \phi(x) \phi(y) \quad (7-1)$$

$$\psi^H(x, y) = \psi(x) \phi(y) \quad (7-2)$$

$$\psi^V(x, y) = \phi(x) \psi(y) \quad (7-3)$$

$$\psi^D(x, y) = \psi(x) \psi(y) \quad (7-4)$$



沿着不同的方向,小波函数会有变化,  $\psi^H$  度量沿着列变化(如水平边缘),  $\psi^V$  度量沿着行变化(如垂直边缘),  $\psi^D$  则对应于对角线方向。每个小波上的  $H$  表示水平方向,  $V$  表示垂直方向,  $D$  表示对角线方向。

由式(7-1)~式(7-4)给出的尺度函数和小波函数,可以定义一个伸缩和平移的基函数

$$\phi_{j,m,n}(x,y) = 2^{j/2} \phi(2^j x - m, 2^j y - n) = \phi_{j,m}(x) \phi_{j,n}(y) \quad (7-5)$$

$$\psi_{j,m,n}^H(x,y) = 2^{j/2} \psi^H(2^j x - m, 2^j y - n) = \psi_{j,m}(x) \phi_{j,n}(y) \quad (7-6)$$

$$\psi_{j,m,n}^V(x,y) = 2^{j/2} \psi^V(2^j x - m, 2^j y - n) = \phi_{j,m}(x) \psi_{j,n}(y) \quad (7-7)$$

$$\psi_{j,m,n}^D(x,y) = 2^{j/2} \psi^D(2^j x - m, 2^j y - n) = \psi_{j,m}(x) \psi_{j,n}(y) \quad (7-8)$$

因为  $\{V_j\}_{j \in \mathbb{Z}}$  是一个二维多分辨分析,所以有

$$V_{k+1} = V_k + W_k$$

其中,  $W_k = W_k^H + W_k^V + W_k^D$ 。

任意给定大小为  $M \times N$  的图像  $f(x,y) \in L^2(R)$ ,  $f_N(x,y)$  是  $f$  在  $V_N$  中的投影。这时,对  $f_k(x,y) \in V_k$ 、 $g_k(x,y) \in W_k$  有

$$f_{k+1}(x,y) = f_k(x,y) + g_k(x,y)$$

其中,  $g_k(x,y)$  为  $g_k = g_k^H + g_k^V + g_k^D$ 。

设  $\{a_{l,j}\}, \{b_{l,j}^I\} (I=H,V,D)$  是由两个一元分解序列生成的二元分解序列

$$a_{l,j} = a_l a_j, \quad b_{l,j}^H = b_l a_j, \quad b_{l,j}^V = a_l b_j, \quad b_{l,j}^D = b_l b_j$$

因为  $f_k(x,y) \in V_k$ ,  $g_k(x,y) \in W_k$ , 而  $\{\phi(2^k x - m, 2^k y - n)\}, \{\psi^I(2^k x - m, 2^k y - n)\}$  分别是空间  $V_k$  与  $W_k$  的 Riesz 基, 故  $f_k(x,y) \in V_k$ ,  $g_k(x,y) \in W_k$  能写为

$$\begin{cases} f_k(x,y) = \sum_{m,n} c_{k;n,m} \phi(2^k x - m, 2^k y - n) \\ g_k^I(x,y) = \sum_{m,n} d_{k;m,n}^I \psi^I(2^k x - m, 2^k y - n) \end{cases}$$

可得到分解算法

$$\begin{cases} c_{k;m,n} = \sum_{l,j} a_{l-2m,j-2n} c_{k+1;m,n} \\ d_{k;m,n}^I = \sum_{l,j} b_{l-2m,j-2n}^I c_{k+1;m,n} \end{cases}$$

设  $\{p_{l,j}\}, \{q_{l,j}^I\} (I=H,V,D)$  是由两个一元二尺度序列得到的二元二尺度序列(即重构序列), 即

$$p_{l,j} = p_l p_j, \quad q_{l,j}^H = q_l p_j, \quad q_{l,j}^V = p_l q_j, \quad q_{l,j}^D = q_l q_j$$

则重构算法为:

$$c_{k+1;m,n} = \sum_{l,j} (p_{m-2l,n-2j} c_{k;l,j} + \sum_{i=1}^3 q_{m-2l,n-2j}^i d_{k;l,j}^i)$$

图像的小波编码过程首先是对原始图像实行二维小波变换,得到小波变换系数。由于小波变换能将原始图像的能量集中到少部分小波系数上,且分解后的小波系数在三个方向的细



节分量有高度的局部相关性,这为进一步量化提供了有利条件。因此,应用小波编码可得到较高的压缩比,且压缩速度较快。

## 7.2 小波变换系数分析

如果采用正交小波变换,那么从理论上讲,小波变换前后的总能量是不变的,是一种能量守恒的变换。并且具有一种能量集中的特性,即将图像的能量集中在低频部分,而在各高频子图像仅有很少比例的能量。将子图 ( $M \times N$  个像素) 的能量定义为

$$E = \frac{1}{MN} \sum_j \sum_i |x(i, j)|^2$$

如果采用双正交小波变换,那么尽量采用近似于正交的双正交小波基,使变换后能量尽量保持不变。在编码的量化阶段经常要用到图像的概率密度函数(PDF),小波变换后各子带概率密度函数可以通过统计方法逼近。仿真实验统计结果表明,在高频子带,小波变换系数更符合广义高斯分布,即对于  $m, d$  子带,PDF 可由如下函数逼近:

$$p_{m,d}(x) = a_{m,d} \exp(-|b_{m,d} x|^{r_{m,d}})$$

其中,  $r_{m,d}$  是 PDF 形状控制参数。当  $r_{m,d} = 2$  时,广义高斯分布就变成高斯分布;当  $r_{m,d} = 1$  时,广义高斯分布就变成拉普拉斯分布。 $r_{m,d}$  越小,PDF 的变化越陡。对于小波变换后的系数,除近似子图像 LL,  $r_{m,d} = 0.7$  时能得到实际统计和逼近函数最接近的曲线。

## 7.3 实验结果与分析

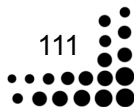
不同于傅里叶分析,小波基不是唯一的,显然难点在于如何选择最优的小波基用于图像压缩,一般情况下需要考虑以下几个因素:

- (1) 小波基的正则性和消失矩。
- (2) 小波基的线性相位。
- (3) 所处理图像与小波基的相似性。
- (4) 小波函数的能量集中性。
- (5) 综合考虑压缩效率和计算复杂度。

正则性是函数光滑性的一种描述,也反映了函数频域能量集中的程度。正则性对图像压缩效果有一定的影响,如果图像大部分是光滑的,一般选择正则性好的小波。如 Haar 小波是不连续的(即不光滑的),会造成复原图像中出现方块效应,而采用其他的小波基则方块效应会消失。

现选择 bior3.7 正交小波对图像进行分解。图 7-1(a) 所示是原始图像;图 7-1(b) 所示是分解后的低频和高频图像;图 7-1(c) 所示是第一次压缩后图像;图 7-1(d) 所示是第二次压缩后图像。原始图像的大小为 524288 B;第一次压缩后图像的大小为 131072 B;第二次压缩后图像的大小为 32768 B。

第一次压缩提取原始图像中小波分解第一层的低频信息,此时压缩效果较好,压缩比较







小，约为  $1/4$ ；第二次压缩是提取第一层分解低频部分的低频部分，即第二低频部分，其压缩比较大，约为  $1/16$ ，压缩效果在视觉上也基本过得去。随着分解层数的增加，压缩比是递减的。

### 7.3.1 小波变换的图像压缩

【例 7-1】基于小波变换的图像压缩。

```
%装载并显示原始图像
load facets;
figure
image(X);
colormap(map);
title('(a)原始图像 ');
axis square;
disp('压缩前图像的大小：');
whos('X')

%对图像进行 7 层小波分解
[c,l]=wavedec2(X,2,'bior3.7');
%提取小波分解结构中第一层的低频系数和高频系数
cA1=appcoef2(c,l,'bior3.7',1);
%水平方向
cH1=detcoef2('h',c,l,1);
%斜线方向
cD1=detcoef2('d',c,l,1);
%垂直方向
cV1=detcoef2('v',c,l,1);

%重构第一层系数
A1=wrcoef2('a',c,l,'bior3.7',1);
H1=wrcoef2('h',c,l,'bior3.7',1);
D1=wrcoef2('d',c,l,'bior3.7',1);
V1=wrcoef2('v',c,l,'bior3.7',1);
c1=[A1 H1;V1 D1];

%显示第一层频率信息
figure
image(c1);
title('(b)分解后的低频和高频信息');

%对图像进行压缩：保留第一层低频信息并对其进行量化编码
ca1=wcodemat(cA1,440,'mat',0);
```

```
%改变图像高度并显示
ca1=0.1*ca1;
figure
image(ca1);
colormap(map);
title('(c)第一次压缩后图像');
axis square;
disp('第一次压缩后图像的大小：');
whos('ca1')
%压缩图像：保留第二层低频信息并对其进行量化编码
cA2=appcoef2(c,l,'bior3.7',2);
ca2=wcodemat(cA2,440,'mat',0);
ca2=0.1*ca2;
figure
image(ca2);
colormap(map);
title('(d)第二次压缩后图像');
disp('第二次压缩后图像大小：');
whos('ca2')
```

程序运行结果如图 7-1 所示。

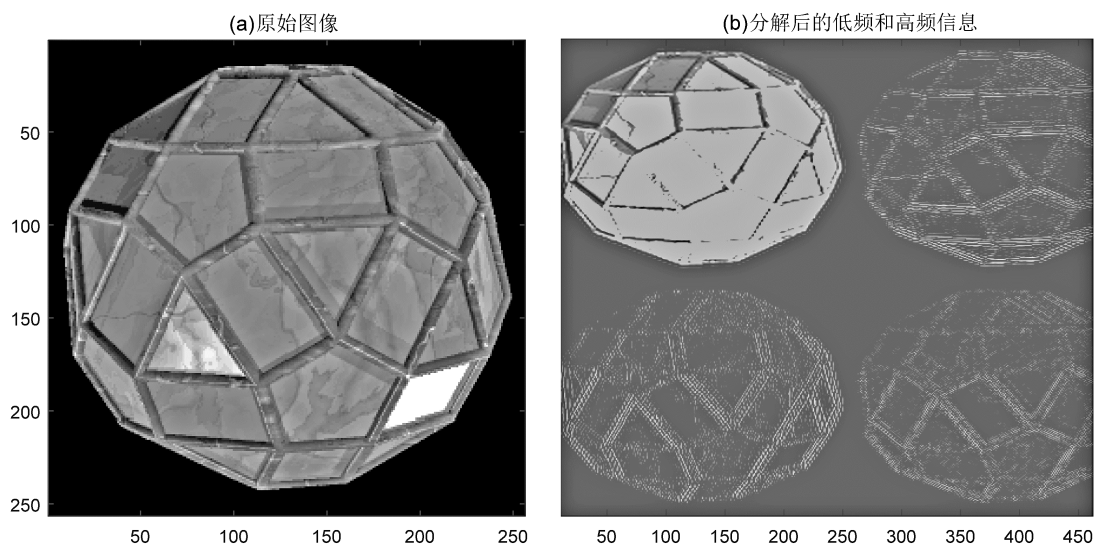


图 7-1 基于小波变换的图像压缩结果

保留原始图像中低频信息的压缩方法只是一种最简单的压缩方法。它不需要经过其他处理即可获得较好的压缩效果。当然，对于上面的例子我们还可以提取小波分解的更高层的低频信息。从理论上说，通过小波变换可以获得任意压缩比的压缩图像，只不过在对压缩比和图像质量都有较高要求时，它就不如其他压缩方法了。

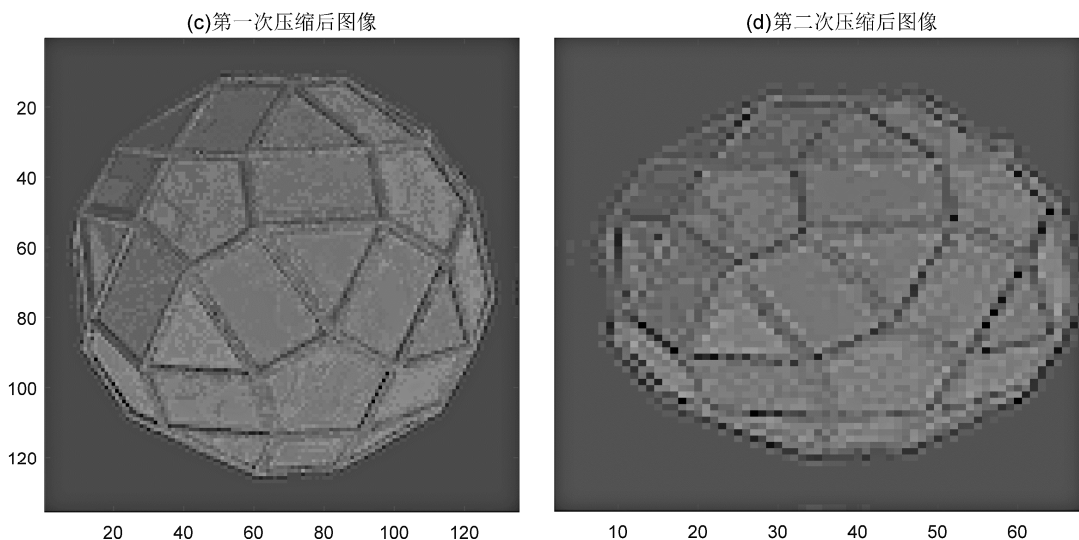


图 7-1 基于小波变换的图像压缩结果（续）

### 7.3.2 sym8 小波对图像进行分解

【例 7-2】采用 sym8 正交小波对 cameraman.tif 图像进行 3 次分解。

```
gray=imread('cameraman.tif');  
[X,map]=gray2ind(gray);  
nbcol=size(map,1);  
figure(14);  
image(X);colormap(map);  
[c,s]=wavedec2(X,3,'sym8');  
A1=wrcoef2('a',c,s,'sym8',1);  
H1=wrcoef2('h',c,s,'sym8',1);  
V1=wrcoef2('v',c,s,'sym8',1);  
D1=wrcoef2('d',c,s,'sym8',1);  
A2=wrcoef2('a',c,s,'sym8',2);  
H2=wrcoef2('h',c,s,'sym8',2);  
V2=wrcoef2('v',c,s,'sym8',2);  
D2=wrcoef2('d',c,s,'sym8',2);  
A3=wrcoef2('a',c,s,'sym8',3);  
H3=wrcoef2('h',c,s,'sym8',3);  
V3=wrcoef2('v',c,s,'sym8',3);  
D3=wrcoef2('d',c,s,'sym8',3);
```

```
figure(1)
```



```
image(wcodemat(A1,nbcol));colormap(map);
title('(a) 尺度为 1 时的低频图像');
figure(2)
image(wcodemat(H1,nbcol));colormap(map);
title('(b) 尺度为 1 时的水平高频图像');
figure(3)
image(wcodemat(V1*90,nbcol));colormap(map);
title('(c) 尺度为 1 时的垂直高频图像');
figure(4)
image(wcodemat(D1*89,nbcol));colormap(map);
title('(d) 尺度为 1 时的对角高频图像');

figure(5)
image(wcodemat(A2,nbcol));colormap(map);
title('(e) 尺度为 2 时的低频图像');
figure(6)
image(wcodemat(H2,nbcol));colormap(map);
title('(f) 尺度为 2 时的水平高频图像');
figure(7)
image(wcodemat(V2,nbcol));colormap(map);
title('(g) 尺度为 2 时的垂直高频图像');
figure(8)
image(wcodemat(D2,nbcol));colormap(map);
title('(h) 尺度为 2 时的对角高频图像');

figure(9)
image(wcodemat(A3,nbcol));colormap(map);
title('(i) 尺度为 3 时的低频图像');
figure(10)
image(wcodemat(H3,nbcol));colormap(map);
title('(j) 尺度为 3 时的水平高频图像');
figure(11)
image(wcodemat(V3,nbcol));colormap(map);
title('(k) 尺度为 3 时的垂直高频图像');
figure(12)
image(wcodemat(D3,nbcol));colormap(map);
title('(l) 尺度为 3 时的对角高频图像');
```

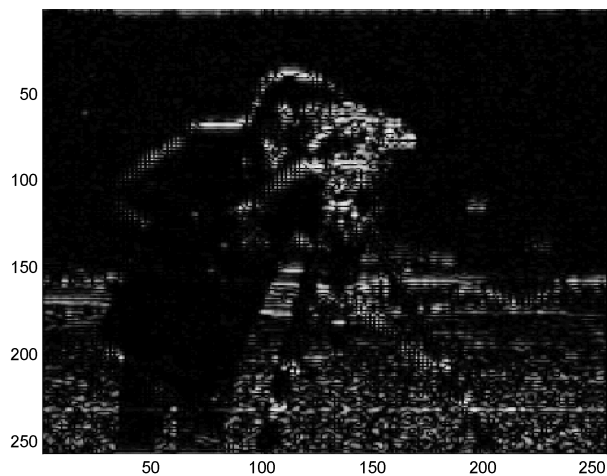
程序运行结果如图 7-2 所示。



(a) 尺度为1时的低频图像



(b) 尺度为1时的水平高频图像



(c) 尺度为1时的垂直高频图像

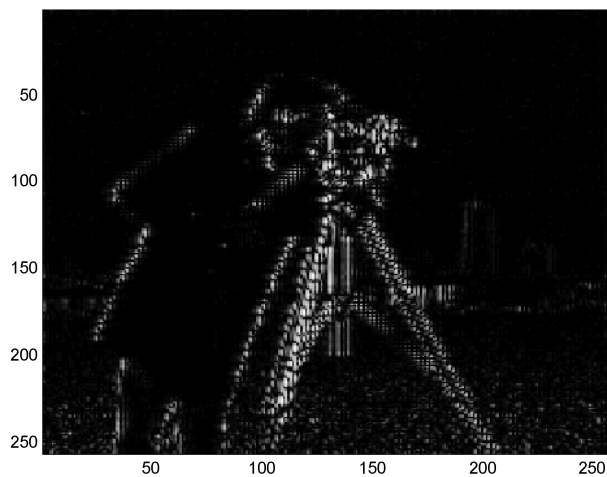
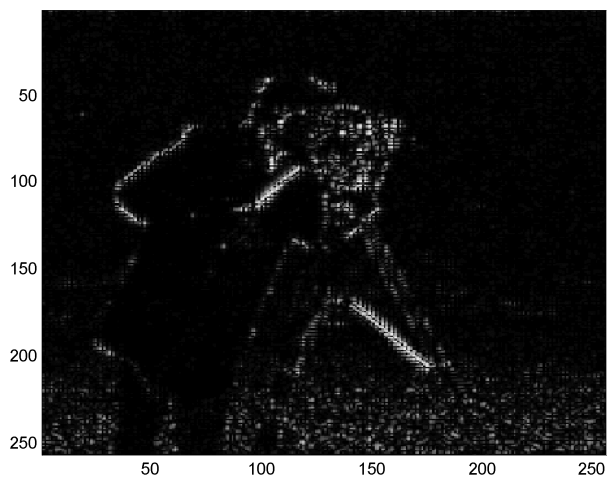
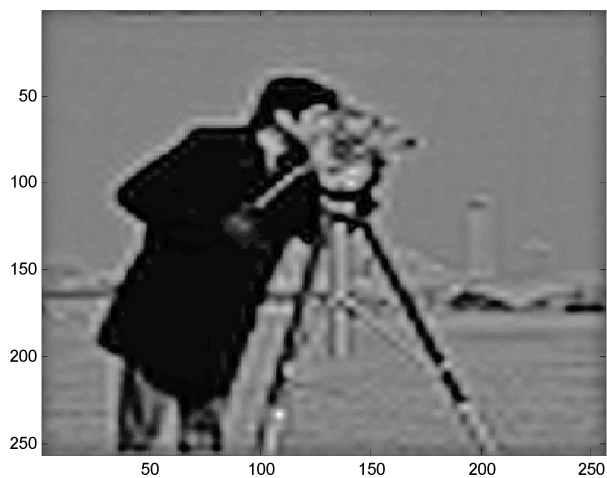


图 7-2 3 次分解后得到的低频和高频图像

(d) 尺度为1时的对角高频图像



(e) 尺度为2时的低频图像



(f) 尺度为2时的水平高频图像

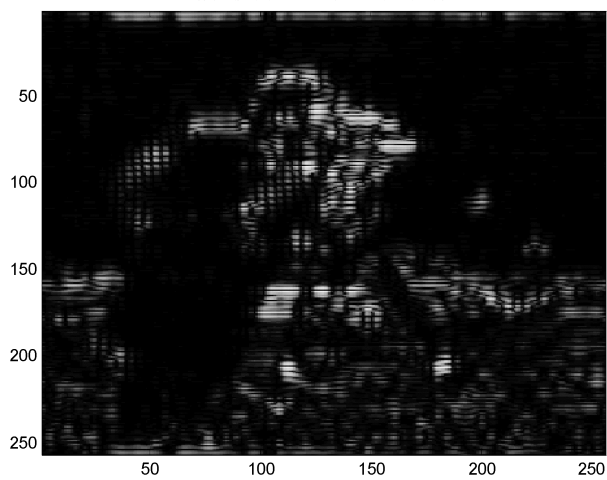
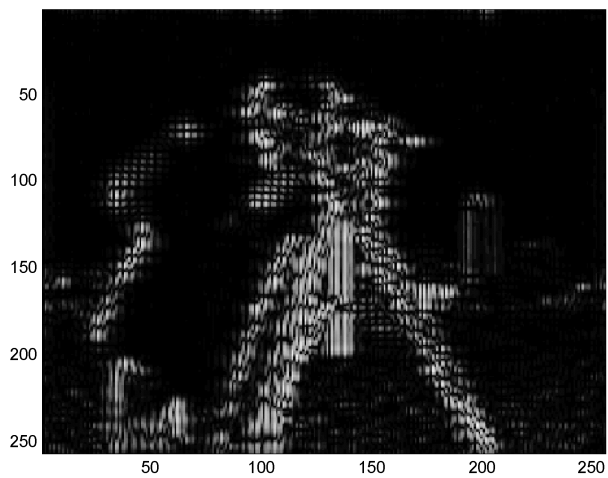


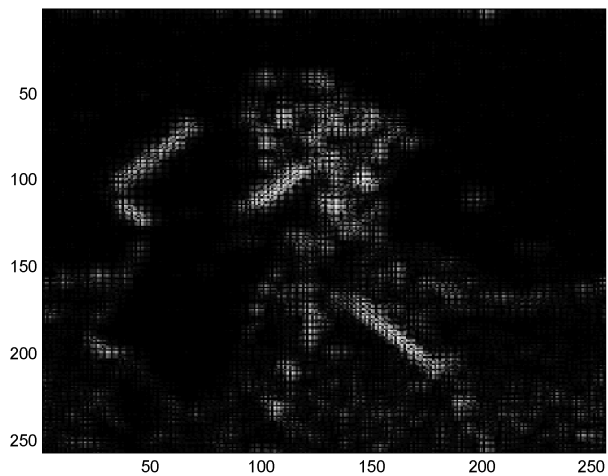
图 7-2 3 次分解后得到的低频和高频图像 (续一)



(g) 尺度为2时的垂直高频图像



(h) 尺度为2时的对角高频图像



(i) 尺度为3时的低频图像

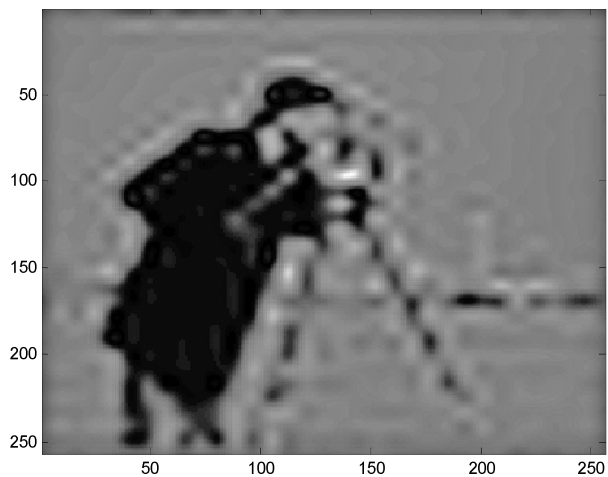
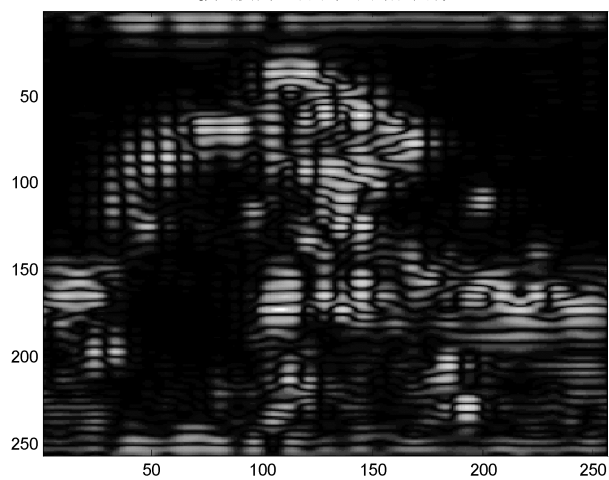
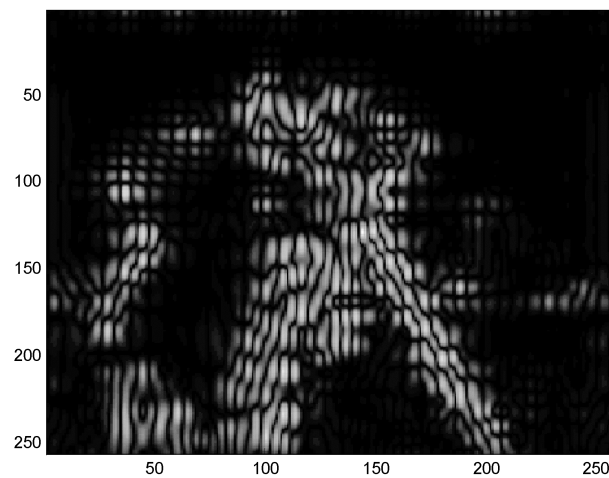


图 7-2 3 次分解后得到的低频和高频图像 (续二)

(j) 尺度为3时的水平高频图像



(k) 尺度为3时的垂直高频图像



(l) 尺度为3时的对角高频图像

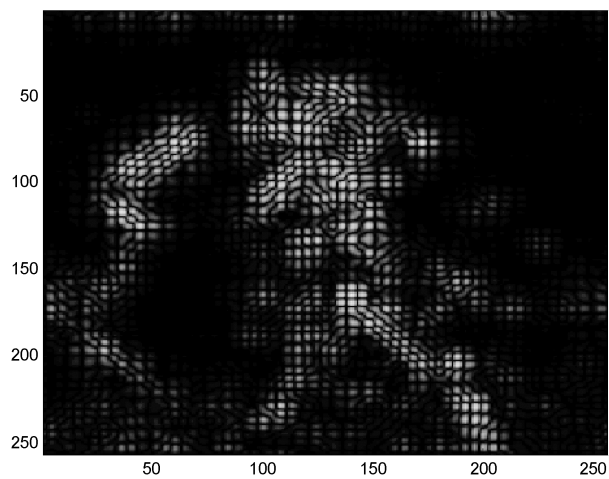


图 7-2 3 次分解后得到的低频和高频图像 (续三)





### 7.3.3 小波系数分布理论分析

【例 7-3】验证小波系数分布理论。

```
Io=imread('cameraman.tif');
I=double(Io);
wname='sym8';
dwtmode('per');
%%小波分解
[I11 IH1 IV1 ID1]=dwt2(I,wname);
[I21 IH2 IV2 ID2]=dwt2(I11,wname);
[I31 IH3 IV3 ID3]=dwt2(I21,wname);
window=[7 7];
Opname='std2';
%%计算方差
sigmI11=nlfilter(I11,window,Opname);
sigmIH1=nlfilter(IH1,window,Opname);
sigmIV1=nlfilter(IV1,window,Opname);
sigmID1=nlfilter(ID1,window,Opname);
sigmI21=nlfilter(I21,window,Opname);
sigmIH2=nlfilter(IH2,window,Opname);
sigmIV2=nlfilter(IV2,window,Opname);
sigmID2=nlfilter(ID2,window,Opname);
sigmI31=nlfilter(I31,window,Opname);
sigmIH3=nlfilter(IH3,window,Opname);
sigmIV3=nlfilter(IV3,window,Opname);
sigmID3=nlfilter(ID3,window,Opname);
%%均值
Opname='mean2';
meanI11=nlfilter(I11,window,Opname);
meanIH1=nlfilter(IH1,window,Opname);
meanIV1=nlfilter(IV1,window,Opname);
meanID1=nlfilter(ID1,window,Opname);

meanI21=nlfilter(I21,window,Opname);
meanIH2=nlfilter(IH2,window,Opname);
meanIV2=nlfilter(IV2,window,Opname);
meanID2=nlfilter(ID2,window,Opname);

meanI31=nlfilter(I31,window,Opname);
meanIH3=nlfilter(IH3,window,Opname);
meanIV3=nlfilter(IV3,window,Opname);
meanID3=nlfilter(ID3,window,Opname);
```



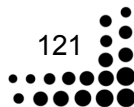
```

IsI11=I11./(sigmI11+.1^10);
IsIH1=IH1./(sigmIH1+.1^10);
IsIV1=IV1./(sigmIV1+.1^10);
IsID1=ID1./(sigmID1+.1^10);

IsI21=I21./(sigmI21+.1^10);
IsIH2=IH2./(sigmIH2+.1^10);
IsIV2=IV2./(sigmIV2+.1^10);
IsID2=ID2./(sigmID2+.1^10);

IsI31=I31./(sigmI31+.1^10);
IsIH3=IH3./(sigmIH3+.1^10);
IsIV3=IV3./(sigmIV3+.1^10);
IsID3=ID3./(sigmID3+.1^10);
S1=size(IsI11);
Iss1=reshape(IsI11,S1(1)*S1(2),1);
Iss2=reshape(IsIH1,S1(1)*S1(2),1);
Iss3=reshape(IsIV1,S1(1)*S1(2),1);
Iss4=reshape(IsID1,S1(1)*S1(2),1);
S2=size(IsI21);
Iss21=reshape(IsI21,S2(1)*S2(2),1);
Iss22=reshape(IsIH2,S2(1)*S2(2),1);
Iss23=reshape(IsIV2,S2(1)*S2(2),1);
Iss24=reshape(IsID2,S2(1)*S2(2),1);
S3=size(IsI31);
Iss31=reshape(IsI31,S3(1)*S3(2),1);
Iss32=reshape(IsIH3,S3(1)*S3(2),1);
Iss33=reshape(IsIV3,S3(1)*S3(2),1);
Iss34=reshape(IsID3,S3(1)*S3(2),1);
%%显示分布图
figure(1);
histfit(Iss1);
title('(a) 尺度为 1 时低频系数分布');
figure(2);
histfit(Iss2);
title('(b) 尺度为 1 时水平高频系数分布');
figure(3);
histfit(Iss3);
title('(c) 尺度为 1 时垂直高频系数分布');
figure(4);
histfit(Iss4);
title('(d) 尺度为 1 时对角高频系数分布');
figure(5);

```





```
histfit(Iss21);  
title('(e) 尺度为 2 时低频系数分布');  
figure(6);  
histfit(Iss22);  
title('(f) 尺度为 2 时水平高频系数分布');  
figure(7);  
histfit(Iss23);  
title('(g) 尺度为 2 时垂直高频系数分布');  
figure(8);  
histfit(Iss24);  
title('(h) 尺度为 2 时对角高频直方图');  
figure(9);  
histfit(Iss31);  
title('(i) 尺度为 3 时低频系数分布');  
figure(10);  
histfit(Iss32);  
title('(j) 尺度为 3 时水平高频系数分布');  
figure(11);  
histfit(Iss33);  
title('(k) 尺度为 3 时垂直高频系数分布');  
figure(12);  
histfit(Iss34);  
title('(l) 尺度为 3 时对角高频系数分布');
```

程序运行结果如图 7-3 所示。

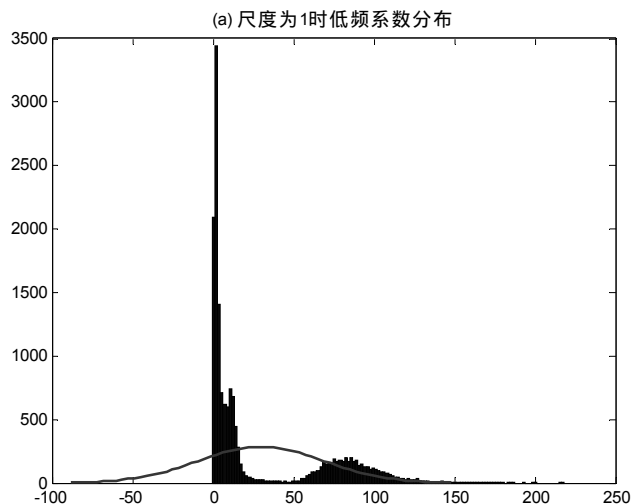


图 7-3 图像的 3 层系数分布

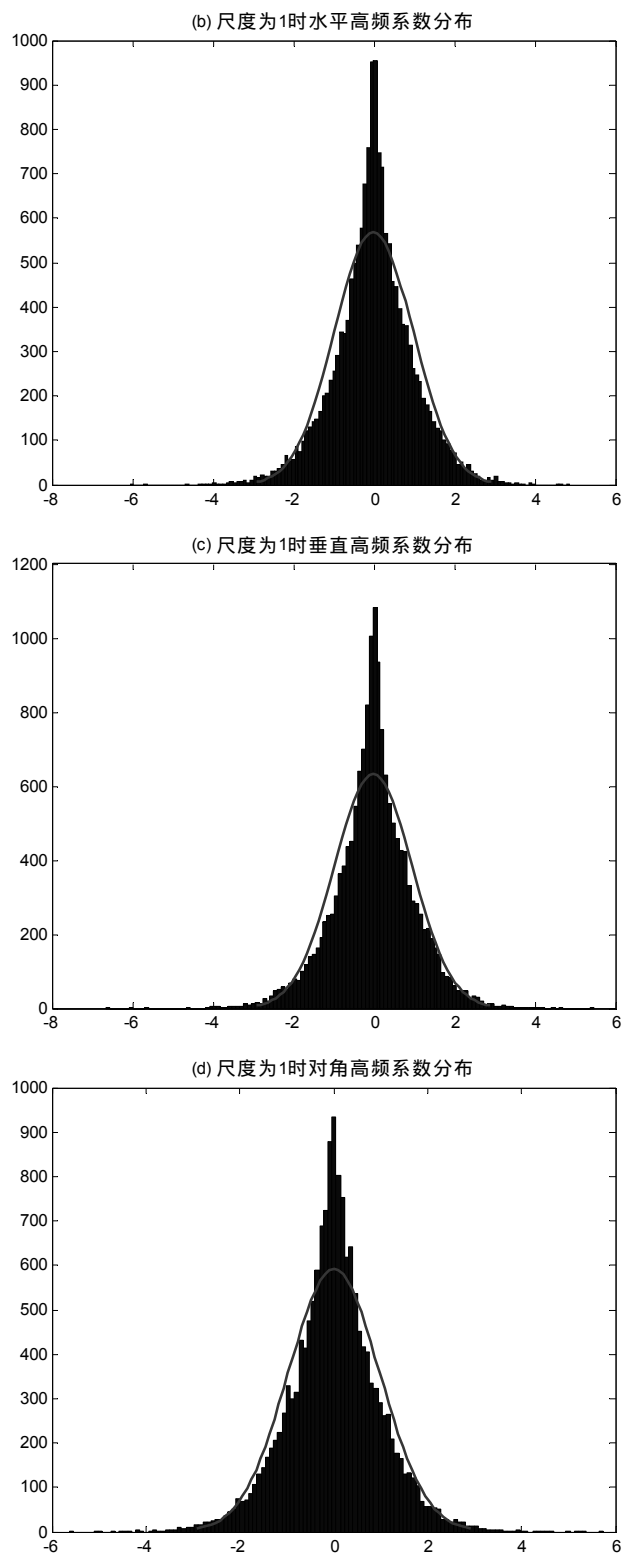


图 7-3 图像的3层系数分布 (续一)

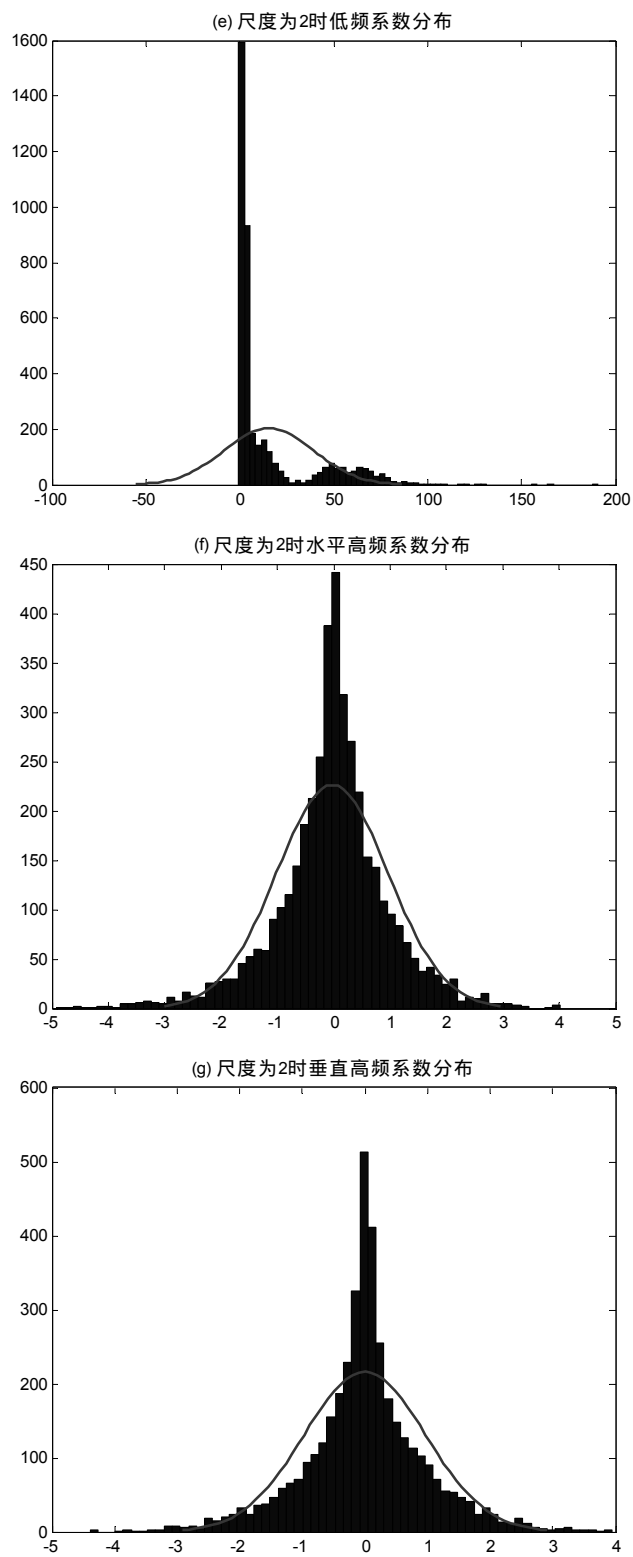


图 7-3 图像的 3 层系数分布 (续三)

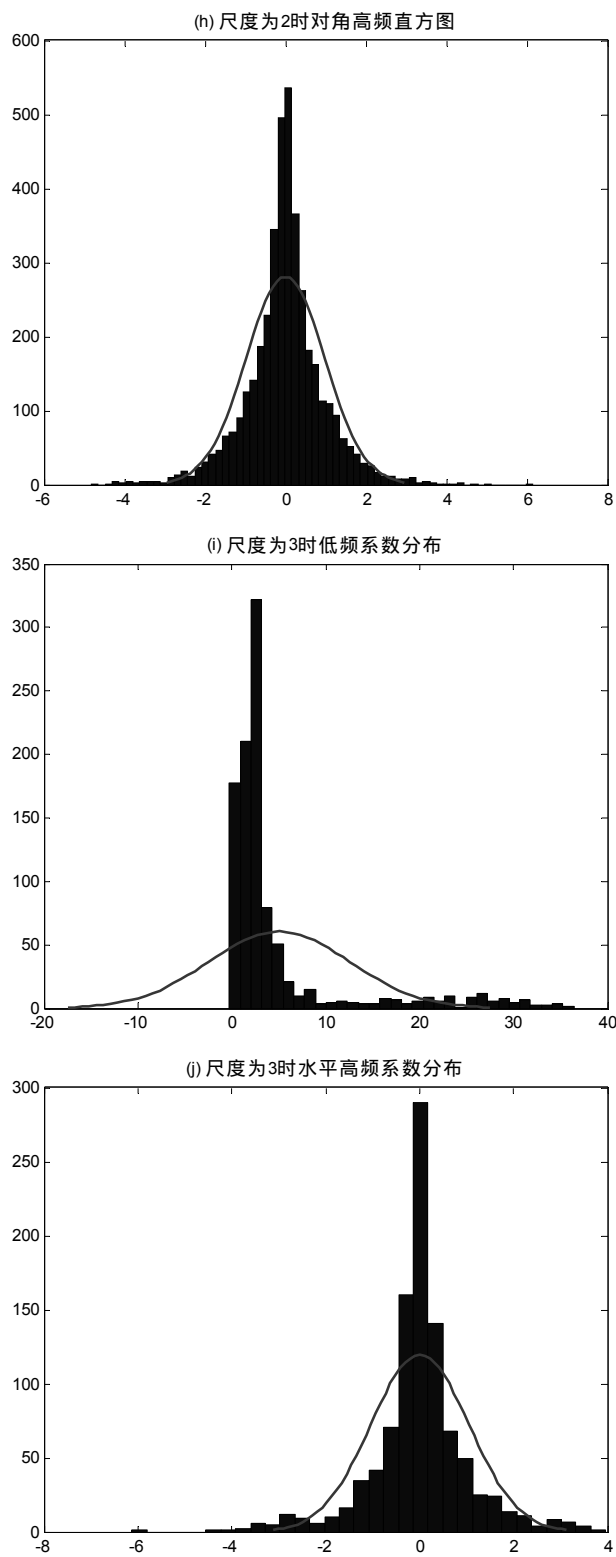
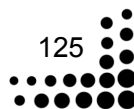


图 7-3 图像的3层系数分布 (续三)



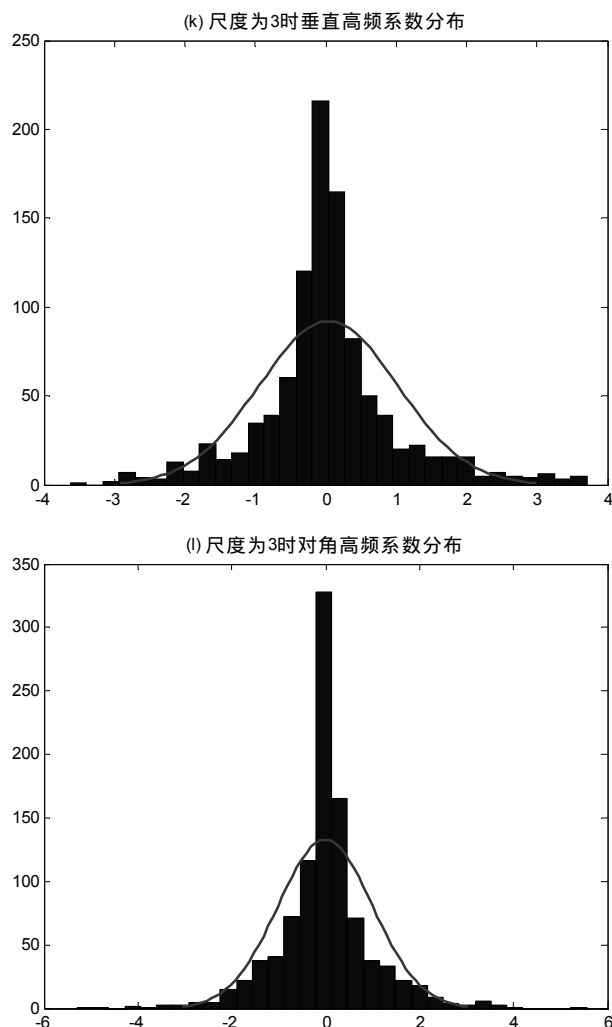


图 7-3 图像的 3 层系数分布 (续四)

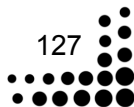
图 7-2 所示是采用 sym8 正交小波对原图像进行 3 次分解后得到的低频和高频图像。图 7-3 所示是图 7-2 所对应的小波系数的分布直方图。根据中心极限定律，在大样本的情况下，小波系数相对于局部的方差应该服从正态分布，小波系数的均值为零。图 7-3 是仿真程序的运行结果，显示了小波系数的直方图以及相应的近似正态分布曲线，是为了验证小波系数的能量分布情况。从图中可以看出，在尺度为 1 的情况下，低通小波系数不服从正态分布而高通小波系数很好地拟合正态分布；在尺度为 2 和 3 的情况下，低通小波系数和高通小波系数都不服从正态分布。原因是第二层和第三层小波系数数目少，不能够很好地拟合大数定理。

(1) 一般情况下，正则性阶数越高对图像压缩效果越好，但在某些情况下也存在相反的情况，即正则性阶数小的小波基反而对图像的压缩效果好。这主要是因为所采用的小波函数与待压缩图像结构上存在一定的相似性。例如，由电视图像信号发生器产生的棋盘信号和方格信号，在相同数码下，用 Haar 小波基压缩的效果就好于用 Daubechies 小波 ( $N>2$ ) 得到的



效果，其原因是 Haar 小波函数的基本图像与原图像的结构有一定的相似性。

(2) 对图像做小波分解后，可得到一系列不同的分辨率的子图像。而对于图像来说，表征它最主要部分是低频部分，而大部分高频点的数值均接近于 0，而且频率越高，这种现象越明显。因此，利用小波分解去掉图像的高频部分而仅保留图像的低频部分是一种最简单的图像压缩方法。





## 第 8 章 提升小波变换的 MATLAB 实现



### 8.1 MATLAB 一维提升小波变换

与传统的小波分解相比，提升小波可以实现整数小波变换，这对于许多应用而言是非常重要的性质。

#### 8.1.1 一维信号压缩 wdcbm 函数应用

**【例 8-1】**一维信号压缩，使用函数 wdcbm 获取信号压缩阈值。

本例中首先对信号进行提升小波分解，然后构造传统小波分解结构[c,l]，接着使用函数 wdcbm 获取信号压缩阈值，最后采用函数 wdencmp 实现信号压缩。

程序清单如下：

```
load nelec;
indx=1:1024;
x=nelec(indx);
% 得到 Haar 小波的提升方案
lshaar=liftwave('haar');
% 将提升步骤 ELS 添加到提升方案中
els={'p',[-0.125 0.125],0};
lsnew=addlift(lshaar,els);
% 进行单层提升小波分解
[cA,cD]=lwt(x,lsnew);
length=size(cA,2);
c=zeros(1,length*2);
for i=1:length;
    c(i)=cA(i);
end
for i=length+1:2*length;
    c(i)=cD(i-length);
end;

l(1)=length;
```



```
l(2)=length;
l(3)=length*2;
alpha=1.5;
% 获得信号压缩的阈值
[thr,nkeep]=wdbcmb(c,l,alpha);
% 对信号进行压缩
[xd,cxd,lxd,perf0,perf12]=wdencmp('lvd',c,l,'haar',1,thr,'s');
subplot(211);plot(indx,x);
title('原始信号');
subplot(212);plot(indx,xd);
title('压缩后的信号');
```

程序运行结果如图 8-1 所示。

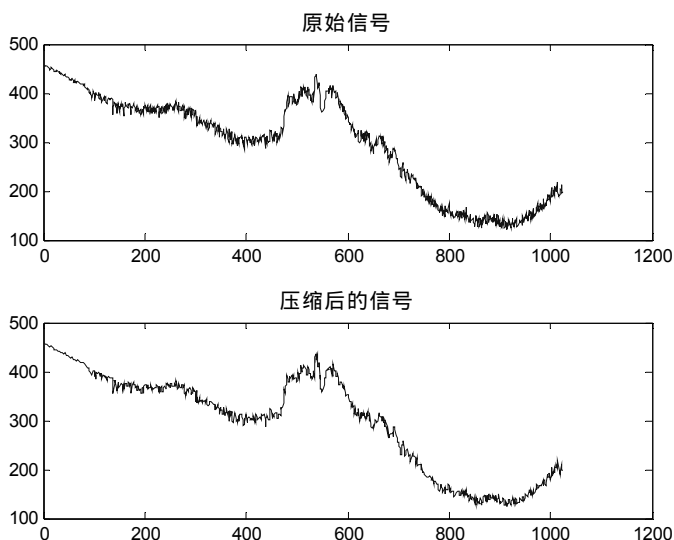


图 8-1 提升分解压缩

信号通过提升小波分解为高频信号和低频信号，高频信号表现为细节信号，通过对高频信号进行处理可以实现信号的压缩。

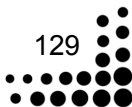
### 8.1.2 一维信号压缩 ddencmp 函数应用

**【例 8-2】**一维信号压缩，使用函数 ddencmp 获取信号压缩阈值。

本例中首先对信号进行提升小波分解，然后构造传统小波分解结构[c,l]，接着使用函数 ddencmp 获取信号压缩阈值，最后采用函数 wdencmp 实现信号压缩。

程序清单如下：

```
load nelec;
indx=1:1024;
x=nelec(indx);
```





```
% 得到 Haar 小波的提升方案
lshaar=liftwave('haar');
% 将提升步骤 ELS 添加到提升方案中
els={'p',[-0.125 0.125],0};
lsnew=addlift(lshaar,els);
% 进行单层提升小波分解
[cA,cD]=lwt(x,lsnew);
length=size(cA,2);
c=zeros(1,length*2);
for i=1:length;
    c(i)=cA(i);
end
for i=length+1:2*length;
    c(i)=cD(i-length);
end;
l(1)=length;
l(2)=length;
l(3)=length*2;
% 获得信号压缩的阈值
[thr,nkeep]=ddencmp('cmp','wv',x);
% 对信号进行压缩
xd=wdencmp('gbl',c,l,'haar',1,thr,'s',1);
subplot(211);plot(indx,x);
title('原始信号');
subplot(212);plot(indx,xd);
title('压缩后的信号');
```

程序运行结果如图 8-2 所示。

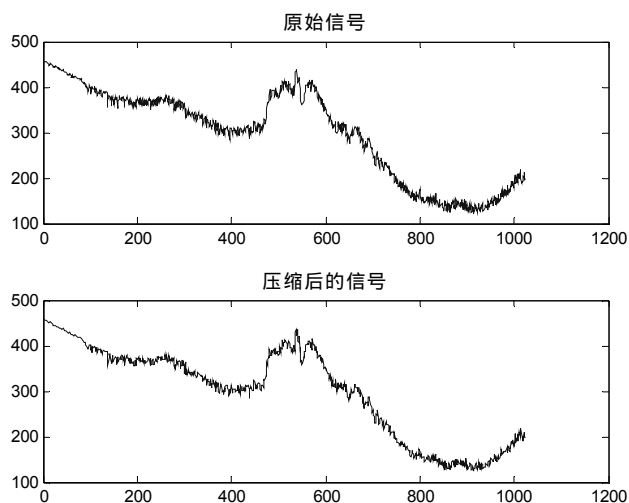


图 8-2 提升分解压缩



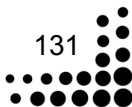
### 8.1.3 信号去噪

#### 【例 8-3】信号去噪。

本例中首先对信号进行两层提升分解，构造小波分解结构 $[c,l]$ ，然后使用函数 `wnoiset` 获取噪声方差，接着使用函数 `wbmpen` 获取小波去噪阈值，最后使用函数 `wdencmp` 实现信号去噪。

程序清单如下：

```
clear all;
clc;
load leleccum;
indx=1:1024;
x=leleccum(indx);
% 产生含噪信号
init=2055615866;
randn('seed',init);
nx=x+18*randn(size(x));
% 得到 Haar 小波的提升方案
lshaar=liftwave('haar');
% 将提升步骤 ELS 添加到提升方案中
els={'p',[-0.125 0.125],0};
lsnew=addlift(lshaar,els);
% 进行提升小波分解
[cA1,cD1]=lwt(x,lsnew);
[cA2,cD2]=lwt(cA1,lsnew);
length=size(cA2,2);
c=zeros(1,length*4);
for i=1:length;
    c(i)=cA2(i);
end
for i=length+1:2*length;
    c(i)=cD2(i-length);
end;
for i=length*2+1:4*length;
    c(i)=cD1(i-2*length);
end;
l(1)=length;
l(2)=length;
l(3)=length*2;
l(4)=length*4;
```





```
% 估计尺度 1 的噪声标准差
sigma=wnoisest(c,l,1);
alpha=2;
% 获得去噪过程中的阈值
thr=wbpenc(c,l,sigma,alpha);
keepapp=1;
% 对信号进行去噪
xd=wdencmp('gbl',c,l,'db6',1,thr,'s',keepapp);
subplot(221);
plot(x);
title('原始信号');
subplot(222);
plot(nx);
title('含噪信号');
subplot(223);
plot(xd(1:1000));
title('去噪后的信号');
```

程序运行结果如图 8-3 所示。

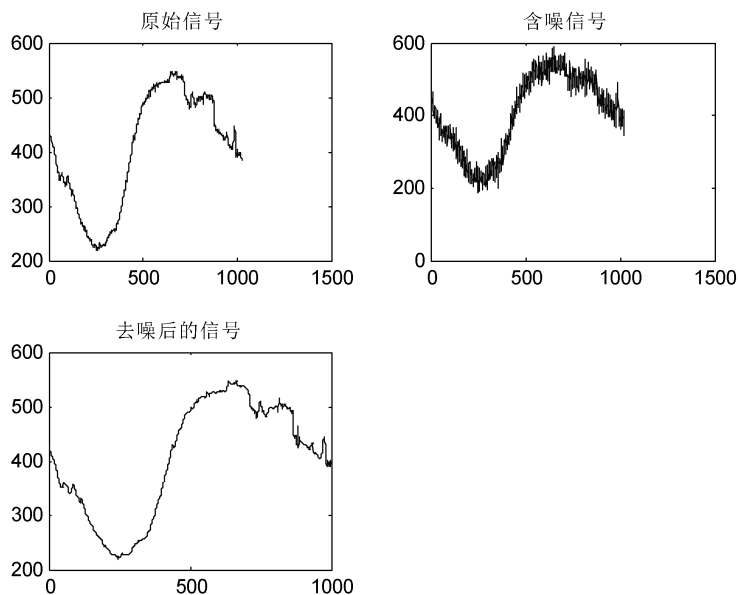


图 8-3 提升分解去噪

本例说明了如何通过信号的提升小波分解进行信号的去噪。其方法与一般去噪方法相同，都是通过对小波分解的高频系数进行阈值量化来达到去噪的目的。

后面将通过 3 个实例说明如何利用 MATLAB 提供的提升的小波函数进行信号的提升分解并提取小波分解的系数。



### 8.1.4 信号的提升分解

【例 8-4】信号的提升分解。

本例中调用函数 `lwt` 进行信号的提升分解。

程序代码如下：

```
% 得到 Haar 小波的提升方案
lshaar=liftwave('haar');
% 添加 ELS 到提升方案中
els={'p',[-0.125 0.125],0};
lsnew=addlift(lshaar,els);
% 进行单层提升小波分解
load noisdopp;
x=noisdopp;
[cA,cD]=lwt(x,lsnew);
figure(1);
subplot(311);
plot(x);
title('原始信号');
subplot(312);
plot(cA);
title('提升小波分解的低频信号');
subplot(313);
plot(cD);
title('提升小波分解的高频信号');
% 直接使用 Haar 小波进行 2 层提升小波分解
[cA,cD]=lwt(x,'haar',2);
figure(2);
subplot(311);
plot(x);
title('原始信号');
subplot(312);
plot(cA);
title('2 层提升小波分解的低频信号');
subplot(313);
plot(cD);
title('2 层提升小波分解的高频信号');
```

程序运行结果如图 8-4 和图 8-5 所示。

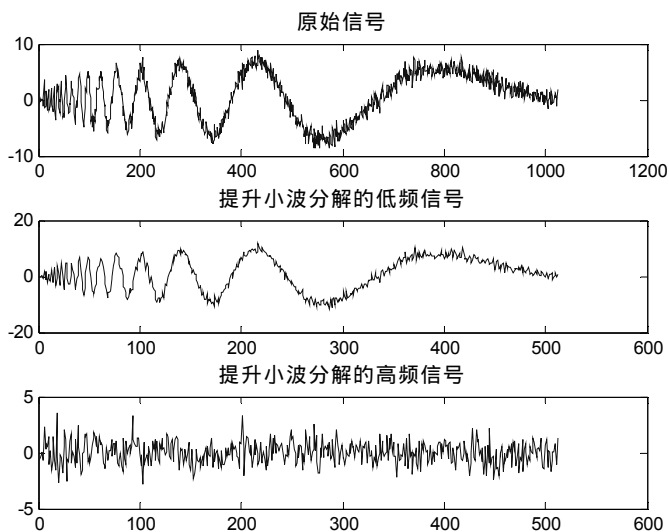


图 8-4 单层提升小波分解

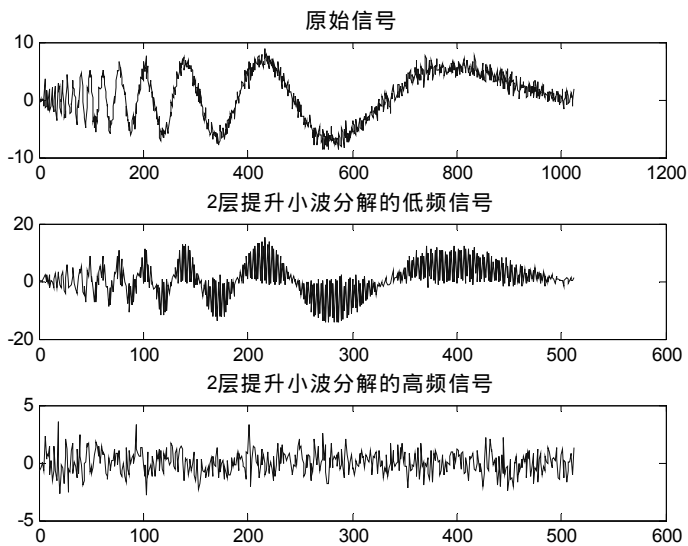


图 8-5 2 层提升小波分解

【例 8-5】提取小波分解的某层小波系数。

程序清单如下：

```
% 得到 Haar 小波的提升方案
lshaar=liftwave('haar');
% 添加 ELS 到提升方案中
els={'p',[-0.125 0.125],0};
lsnew=addlift(lshaar,els);
% 2 层提升小波分解
load noisdopp;
x=noisdopp;
```



```

xDec=lwt(x,lsnew,2);
% 提取第一层的近似系数
ca1=lwtcoef('ca',xDec,lsnew,2,1);
% 提取第二层的近似系数
ca2=lwtcoef('ca',xDec,lsnew,2,2);
% 提取第一层的细节系数
cd1=lwtcoef('cd',xDec,lsnew,2,1);
% 提取第二层的细节系数
cd2=lwtcoef('cd',xDec,lsnew,2,2);
subplot(311);
plot(x);
title('原始信号');
subplot(323);
plot(ca1);
title('第一层近似信号');
subplot(324);
plot(ca2);
title('第二层近似信号');
subplot(325);
plot(cd1);
title('第一层细节信号');
subplot(326);
plot(cd2);
title('第二层细节信号');

```

程序运行结果如图 8-6 所示。

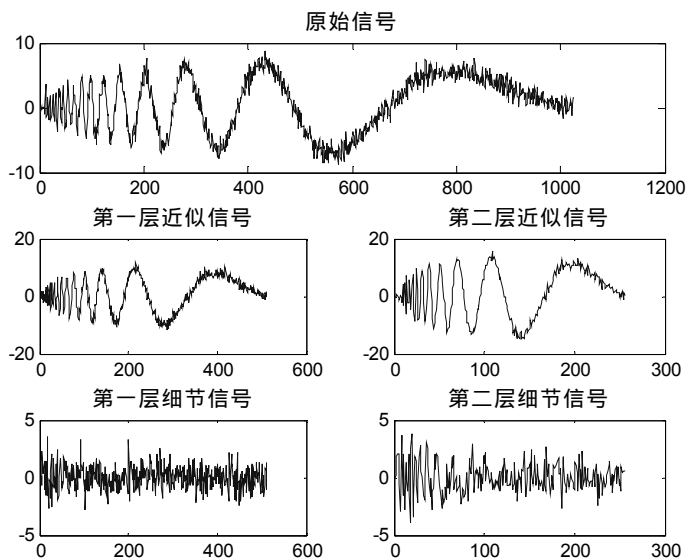
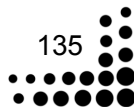


图 8-6 提取提升分解系数







本例演示了如何提取小波分解的某层小波系数，即首先对信号进行提升分解，然后提取信号分解的小波系数。

**【例 8-6】**对信号实施整数提升小波变换。

程序清单如下：

```
% 获得 db2 小波的提升方案
lsdb2=liftwave('db2');
% 显示提升方案
displs(lsdb2);
% 获得 sym2 小波的提升方案
lssym2=liftwave('sym2');
% 显示提升方案
displs(lssym2);
% 获得整数变换提升方案
lsdb2=liftwave('db2','Int2Int');
x=[1:10];
lwtx=lwt(x,lsdb2)
```

程序运行结果如下：

```
lsdb2 = {...
'd'          [-1.73205081]          [0]
'p'          [-0.06698730  0.43301270] [1]
'd'          [ 1.00000000]          [-1]
[ 1.93185165] [ 0.51763809]          []
};
lssym2 = {...
'd'          [-1.73205081]          [0]
'p'          [-0.06698730  0.43301270] [1]
'd'          [ 1.00000000]          [-1]
[ 1.93185165] [ 0.51763809]          []
};
```

```
lwtx =
```

```
1 1 3 0 5 1 6 1 7 1
```

本例演示了如何对信号实施整数提升小波变换。

### 8.1.5 信号的重构

**【例 8-7】**通过提升小波分解的逆变换进行信号的重构。

本例将通过提升小波分解的逆变换进行信号的重构。

程序清单如下：

```
% 获得 Haar 小波的提升方案
```



```

lshaar=liftwave('haar');
% 将提升步骤 ELS 加入到提升方案中
els={'p',[-0.125 0.125],0};
lsnew=addlift(lshaar,els);
% 进行单层提升小波分解
load noisdopp;
x=noisdopp;
subplot(211);
plot(x);
ylabel('x');
% 实施提升小波变换
[cA,cD]=lwt(x,lsnew);
xRec=ilwt(cA,cD,lsnew);
err=max(max(abs(x-xRec)))
subplot(212);
plot(xRec);
ylabel('xRec');

```

程序运行结果如图 8-7 所示。

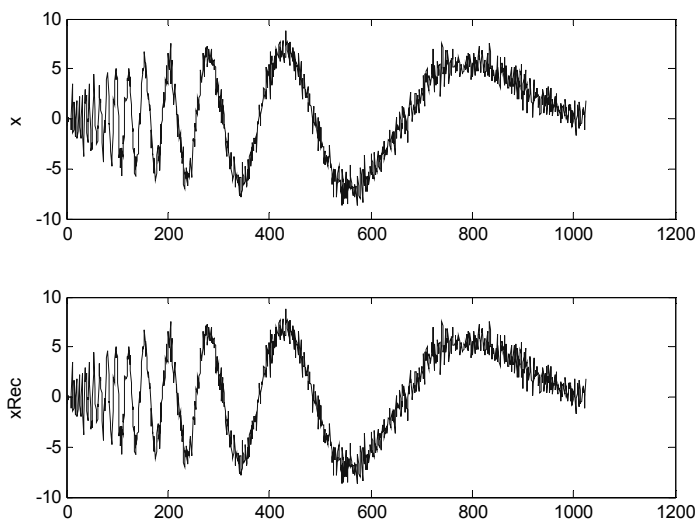


图 8-7 一维提升小波逆变换

计算结果如下：

```

err =
    2.6645e-015

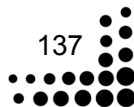
```

**【例 8-8】**对提升小波分解的某一层系数进行单支重构。  
本例演示如何对提升小波分解的某层系数进行单支重构。  
程序清单如下：

```

els={'p',[-0.125 0.125],0};

```





```
lsnew=addlift(lshaar,els);  
% 2 层提升小波分解  
load noisdopp;  
x=noisdopp;  
xDec=lwt(x,lsnew,2);  
% 重构近似信号和细节信号  
a1=lwtcoef('a',xDec,lsnew,2,1);  
a2=lwtcoef('a',xDec,lsnew,2,2);  
d1=lwtcoef('d',xDec,lsnew,2,1);  
d2=lwtcoef('d',xDec,lsnew,2,2);  
% 检查重构误差  
err=max(abs(x-a2-d2-d1))  
subplot(311);  
plot(x);  
title('原始信号');  
subplot(323);  
plot(a1);  
title('重构第一层近似信号');  
subplot(324);  
plot(a2);  
title('重构第二层近似信号');  
subplot(325);  
plot(d1);  
title('重构第一层细节信号');  
subplot(326);  
plot(d2);  
title('重构第二层细节信号');
```

程序运行结果如图 8-8 所示。

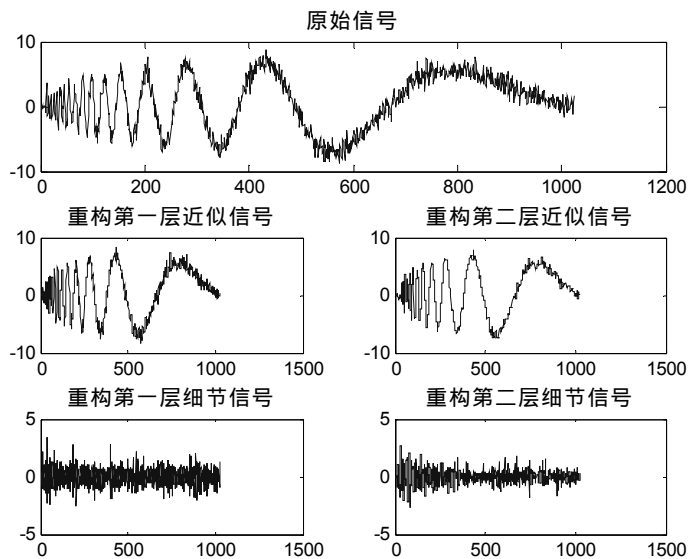


图 8-8 提升小波分解单支重构



计算结果如下：

```
err =  
1.9984e-015
```

**【例 8-9】**对整数小波分解进行重构。

本例说明如何对整数小波分解进行重构。

程序清单如下：

```
% 得到 Haar 小波的提升方案  
lshaar=liftwave('haar');  
% 将提升步骤 ELS 添加到提升方案中  
els={'p',[-0.125 0.125],0};  
lsnew=addlift(lshaar,els);  
% 2 层提升小波分解  
load noisdopp;  
x=noisdopp;  
subplot(211);  
plot(x);  
ylabel('x');  
% 对信号实施整数提升小波变换  
lshaarInt=liftwave('haar','int2int');  
lsnewInt=addlift(lshaarInt,els);  
[cAint,cDint]=lwt(x,lsnewInt);  
% 实施提升小波变换  
xRecInt=ilwt(cAint,cDint,lsnewInt);  
errInt=max(max(abs(x-xRecInt)))  
subplot(212);  
plot(xRecInt);  
ylabel('xRecInt');  
errInt =  
4.4409e-016
```

程序运行结果如图 8-9 所示。

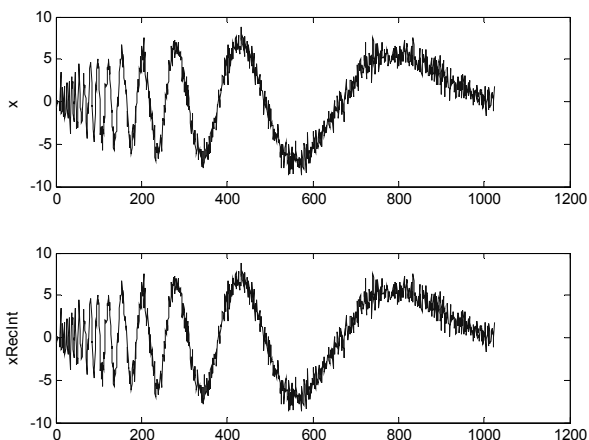
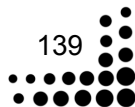


图 8-9 整数提升分解重构



**【例 8-10】lwt 函数应用举例。**

```
% 使用 Haar 小波，得到相应的提升方案
lshaar=liftwave('haar');
% 添加 ELS 到提升方案
els={'p',[-0.125 0.125],0}
lsnew=addlift(lshaar,els);
% 对于简单信号，尺度为 1 进行 LWT
x=1:8;
[cA,cD]=lwt(x,lsnew)
% 对上面的信号，进行整数 LWT
lshaarInt=liftwave('haar','int2int');
lsnewInt=addlift(lshaarInt,els);
[cAint,cDint]=lwt(x,lsnewInt)
```

程序运行结果如下：

```
els =
    'p'    [1x2 double]    [0]
cA =
    1.9445    4.9497    7.7782   10.6066
cD =
    0.7071    0.7071    0.7071    0.7071
cAint =
     1     3     5     7
cDint =
     1     1     1     1
```

**【例 8-11】ilwt 函数应用举例。**

```
% 使用 Haar 小波，得到相应的提升方案
lshaar=liftwave('haar');
% 添加 ELS 到提升方案
els={'p',[-0.125 0.125],0};
lsnew=addlift(lshaar,els);
% 对于简单信号，尺度为 1 进行 LWT
x=1:8;
[cA,cD]=lwt(x,lsnew);
% 对上面的信号，进行整数 LWT
lshaarInt=liftwave('haar','int2int');
lsnewInt=addlift(lshaarInt,els);
[cAint,cDint]=lwt(x,lsnewInt);
% 进行逆变换
xRec=ilwt(cA,cD,lsnew);
err=max(max(abs(x-xRec)))
xRecInt=ilwt(cAint,cDint,lsnewInt);
errInt=max(max(abs(x-xRecInt)))
```



程序运行结果如下：

```
err =  
    4.4409e-016  
errInt =  
    0
```

## 8.2 MATLAB 二维提升小波变换

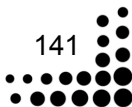
### 8.2.1 图像压缩 wdcbm2 函数应用

【例 8-12】图像压缩实例，使用函数 wdcbm2 获取信号压缩阈值。

本例中首先对二维图像进行提升小波分解，然后构造传统小波分解结构[c,s]，接着使用函数 wdcbm2 获取图像压缩阈值，最后采用函数 wdencomp 实现图像压缩。

程序清单如下：

```
clc;  
load woman;  
nbc=size(map,1);  
% 得到 Haar 小波的提升方案  
lshaar=liftwave('haar');  
% 将提升步骤 ELS 添加到提升方案中  
els={'p',[-0.125 0.125],0};  
lsnew=addlift(lshaar,els);  
load woman;  
% 进行一层提升小波分解  
[cA,cH,cV,cD]=lwt2(X,lsnew);  
length=size(cA,1);  
c=zeros(1,length*length*4);  
for i=1:length;  
    c((i-1)*length+1:i*length)=cA(:,i);  
end;  
for i=length+1:2*length;  
    c((i-1)*length+1:i*length)=cH(:,i-length);  
end;  
for i=2*length+1:3*length;  
    c((i-1)*length+1:i*length)=cV(:,i-2*length);  
end;  
for i=3*length+1:4*length;  
    c((i-1)*length+1:i*length)=cD(:,i-3*length);  
end;  
s=zeros(3,2);  
s(:,1)=[length,length,2*length];
```





```
s(:,2)=[length,length,2*length];  
% 使用 wdcbm2 获得压缩阈值  
alpha=1.5;m=3.5*prod(s(1,:));  
[thr,nkeep]=wdcbm2(c,s,alpha,m);  
% 对图像进行压缩  
xd=wdencmp('lvd',c,s,'haar',1,thr,'h');  
colormap(pink(nbc));  
figure(1);  
subplot(121);  
image(wcodemat(X,nbc));  
title('原始图像');  
subplot(122);  
image(wcodemat(xd,nbc));  
title('压缩后的图像');
```

程序运行结果如图 8-10 所示。

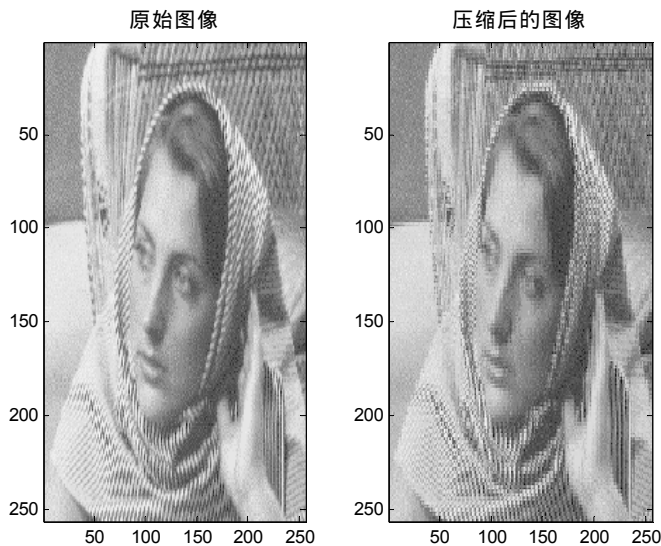


图 8-10 提升分解压缩

## 8.2.2 图像压缩 ddencmp 函数应用

【例 8-13】图像压缩实例，使用函数 ddencmp 获取信号压缩阈值。

本例中首先对二维图像进行提升小波分解，然后构造传统小波分解结构[c,s]，接着使用函数 ddencmp 获取信号压缩阈值，最后采用函数 wdencmp 实现图像压缩。

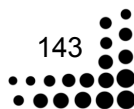
程序清单如下：

```
load woman;  
nbc=size(map,1);
```



```
% 得到 Haar 小波的提升方案
lshaar=liftwave('haar');
% 将提升步骤 ELS 添加到提升方案中
els={'p',[-0.125 0.125],0};
lsnew=addlift(lshaar,els);
load woman;
% 进行 1 层提升小波分解
[cA,cH,cV,cD]=lwt2(X,lsnew);
length=size(cA,1);
c=zeros(1,length*length*4);
for i=1:length;
    c((i-1)*length+1:i*length)=cA(:,i);
end;
for i=length+1:2*length;
    c((i-1)*length+1:i*length)=cH(:,i-length);
end;
for i=2*length+1:3*length;
    c((i-1)*length+1:i*length)=cV(:,i-2*length);
end;
for i=3*length+1:4*length;
    c((i-1)*length+1:i*length)=cD(:,i-3*length);
end;
s=zeros(3,2);
s(:,1)=[length,length,2*length];
s(:,2)=[length,length,2*length];
% 使用 ddencmp 获得压缩阈值
[thr,nkeep]=ddencmp('cmp','wv',X);
% 对图像进行压缩
xd=wdencmp('gbl',c,s,'haar',1,thr,'s',1);
colormap(pink(nbc));
figure(1);
subplot(121);
image(wcodemat(X,nbc));
title('原始图像');
subplot(122);
image(wcodemat(xd,nbc));
title('压缩后的图像');
```

程序运行结果如图 8-11 所示。





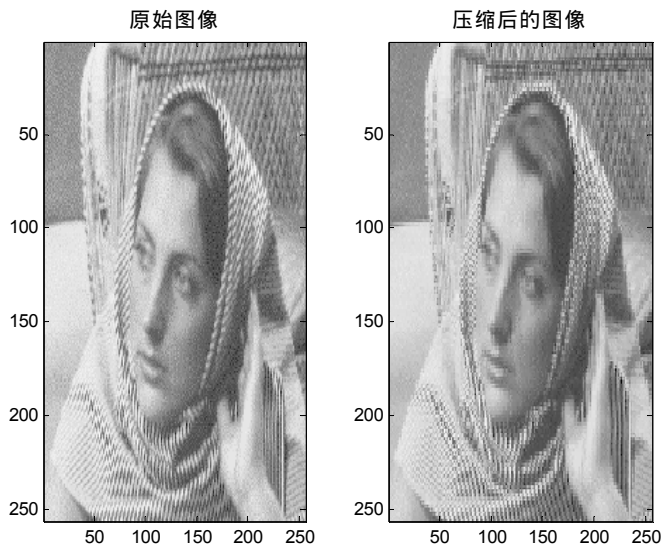


图 8-11 提升分解压缩

### 8.2.3 图像去噪

#### 【例 8-14】图像去噪实例。

本例说明如何通过图像的提升小波分解进行图像的去噪。其方法与一般去噪方法相同，都是通过对小波分解的高频系数进行阈值量化来达到去噪的目的。

本例中首先对图像进行 2 层提升小波分解，然后构造传统小波分解结构[c,s]，接着使用函数 wdcbm2 获得图像去噪阈值，最后使用函数 wdencomp 实现图像去噪。

程序清单如下：

```
load woman;
% 产生含噪图像
init=2055615886;
randn('seed',init);
x=X+18*randn(size(X));
nbc=size(map,1);
% 得到 Haar 小波的提升方案
lshaar=liftwave('haar');
% 将提升步骤 ELS 添加到提升方案中
els={'p',[-0.125 0.125],0};
lsnew=addlift(lshaar,els);
load woman;
% 进行一层提升小波分解
[cA,cH,cV,cD]=lwt2(X,lsnew);
length=size(cA,1);
```



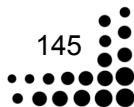
```

c=zeros(1,length*length*4);
for i=1:length;
    c((i-1)*length+1:i*length)=cA(:,i);
end;
for i=length+1:2*length;
    c((i-1)*length+1:i*length)=cH(:,i-length);
end;
for i=2*length+1:3*length;
    c((i-1)*length+1:i*length)=cV(:,i-2*length);
end;
for i=3*length+1:4*length;
    c((i-1)*length+1:i*length)=cD(:,i-3*length);
end;
s=zeros(3,2);
s(:,1)=[length,length,2*length];
s(:,2)=[length,length,2*length];
% 使用 wdcbm2 获得去噪阈值
alpha=3;m=3.5*prod(s(1,:));
[thr,nkeep]=wdcbm2(c,s,alpha,m);
% 对图像进行去噪
sorh='s';
xd=wdencmp('lvd',c,s,'haar',1,thr,sorh);
colormap(pink(nbc));
figure(1);
subplot(221);
image(wcodemat(X,nbc));
title('原始图像');
subplot(222);
image(wcodemat(x,nbc));
title('含噪图像');
subplot(223);
image(wcodemat(xd,nbc));
title('去噪后的图像');

```

程序运行结果如图 8-12 所示。

下面将通过两个例子说明如何利用 MATLAB 提供的提升小波函数进行图像的提升分解并提取小波分解的系数。



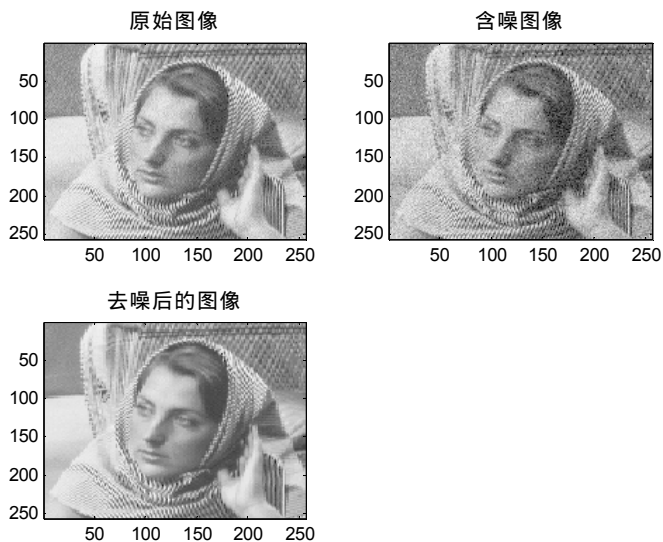


图 8-12 提升分解去噪

## 8.2.4 图像的提升分解

【例 8-15】图像的提升分解。

本例中调用函数 `lwt2` 进行图像的提升分解。

程序清单如下：

```
% 得到 Haar 小波的提升方案
lshaar=liftwave('haar');
% 添加 ELS 到提升方案中
els={'p',[-0.125 0.125],0};
lsnew=addlift(lshaar,els);
load woman;
% 进行 1 层提升小波分解
[cA,cH,cV,cD]=lwt2(X,lsnew);
figure(1);
nbc=size(map,1);
colormap(pink(nbc));
subplot(321);
image(wcodemat(X,nbc));
title('原始图像');
subplot(322);
image(wcodemat(cA,nbc));
title('提升小波分解的低频图像');
subplot(323);
image(wcodemat(cH,nbc));
title('水平方向高频图像');
```



```
subplot(324);
image(wcodemat(cV,nbc));
title('垂直方向高频图像');
subplot(325);
image(wcodemat(cD,nbc));
title('对角方向高频图像');
```

程序运行结果如图 8-13 所示。

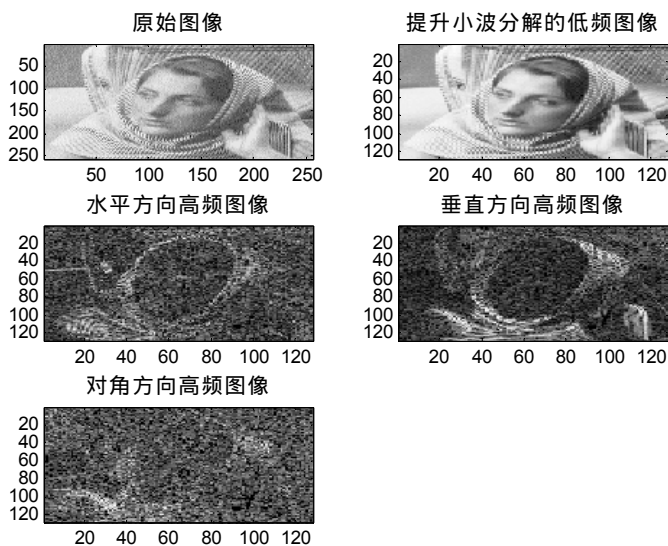


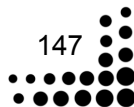
图 8-13 图像提升小波分解

#### 【例 8-16】lwt2 函数应用举例。

```
% 使用 Haar 小波，得到相应的提升方案
lshaar=liftwave('haar');
% 添加 ELS 到提升方案
els={'p',[-0.125 0.125],0}
lsnew=addlift(lshaar,els);
% 对于简单图像，尺度为 1 进行 LWT
x=reshape(1:16,4,4);
[cA,cH,cV,cD]=lwt2(x,lsnew);
% 对上面的图像，进行整数 LWT
lshaarInt=liftwave('haar','int2int');
lsnewInt=addlift(lshaarInt,els);
[cAint,cHint,cVint,cDint]=lwt2(x,lsnewInt)
```

程序运行结果如下：

```
els =
    'p'    [1x2 double]    [0]
cAint =
```





```
3    11
5    13
cHint =
1     1
1     1
cVint =
4     4
4     4
cDint =
0     0
0     0
```

**【例 8-17】**提取小波分解的某层的小波系数。

本例演示如何提取小波分解的某层小波系数，即首先对二维图像进行提升分解，然后提取图像分解的小波系数。

程序清单如下：

```
clear;
clc;
% 得到 Haar 小波的提升方案
lshaar=liftwave('haar');
% 添加 ELS 到提升方案中
els={'p',[-0.125 0.125],0};
lsnew=addlift(lshaar,els);
% 2 层提升小波分解
load woman;
xDec=lwt2(X,lsnew,2);
% 提取近似图像和细节图像
ca2=lwtcoef2('ca',xDec,lsnew,2,2);
ch1=lwtcoef2('ch',xDec,lsnew,2,1);
cv1=lwtcoef2('cv',xDec,lsnew,2,1);
cd1=lwtcoef2('cd',xDec,lsnew,2,1);
ch2=lwtcoef2('ch',xDec,lsnew,2,2);
cv2=lwtcoef2('cv',xDec,lsnew,2,2);
cd2=lwtcoef2('cd',xDec,lsnew,2,2);
nbc=size(map,1);
colormap(pink(nbc));
subplot(121);
image(wcodemat(X,nbc));
title('原始图像');
subplot(122);
image(wcodemat(ca2,nbc));
```



```
title('第二层近似图像');  
figure;  
subplot(231);  
image(wcodemat(ch1,nbc));  
title('第一层水平方向图像');  
subplot(232);  
image(wcodemat(cv1,nbc));  
title('第一层垂直方向图像');  
subplot(233);  
image(wcodemat(cd1,nbc));  
title('第一层对角方向图像');  
subplot(234);  
image(wcodemat(ch2,nbc));  
title('第二层水平方向图像');  
subplot(235);  
image(wcodemat(cv2,nbc));  
title('第二层垂直方向图像');  
subplot(236);  
image(wcodemat(cd2,nbc));  
title('第二层对角方向图像');
```

程序运行结果如图 8-14 和图 8-15 所示。

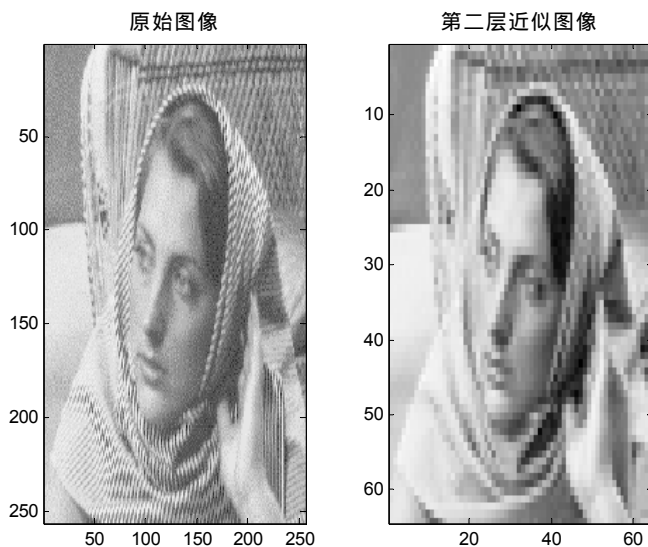
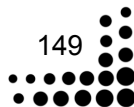


图 8-14 提升分解的低频系数



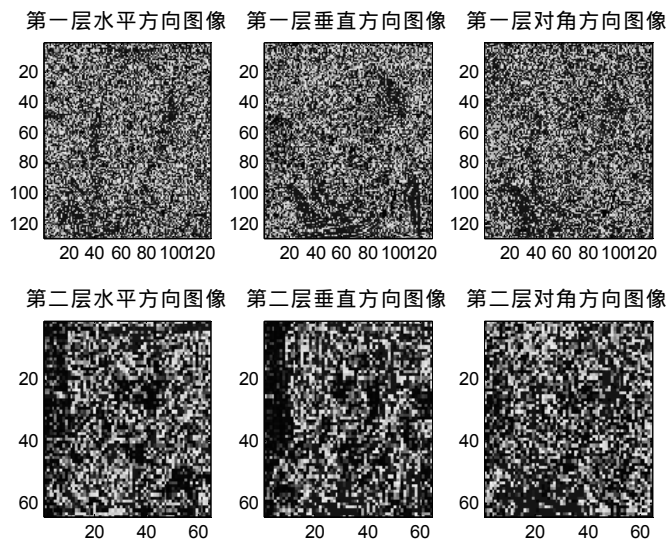


图 8-15 提取提升分解的高频系数

### 8.2.5 图像的提升重构

本小节通过两个例子说明如何在 MATLAB 中进行图像的提升分解重构。

**【例 8-18】**图像重构。

程序清单如下：

```
% 获得 Haar 小波的提升方案
lshaar=liftwave('haar');
% 将提升步骤 ELS 加入到提升方案中
els={'p',[-0.125 0.125],0};
lsnew=addlift(lshaar,els);
% 实施二维提升小波分解
load woman;
nbc=size(map,1);
colormap(pink(nbc));
subplot(221);
image(wcodemat(X,nbc));
title('X');
[cA,cH,cV,cD]=lwt2(X,lsnew);
% 对同一个图像实施整数提升小波变换
lshaarInt=liftwave('haar','int2int');
lsnewInt=addlift(lshaarInt,els);
[cAint,cHint,cVint,cDint]=lwt2(X,lsnewInt);
% 实施提升小波逆变换
xRec=ilwt2(cA,cH,cV,cD,lsnew);
```



```

err=max(max(abs(X-xRec)))
subplot(222);
image(wcodemat(xRec,nbc));
title('xRec');
xRecInt=ilwt2(cAint,cHint,cVint,cDint,lsnewInt);
errInt=max(max(abs(X-xRecInt)))
subplot(223);
image(wcodemat(xRecInt,nbc));
title('xRecInt');

```

程序运行结果如图 8-16 所示。

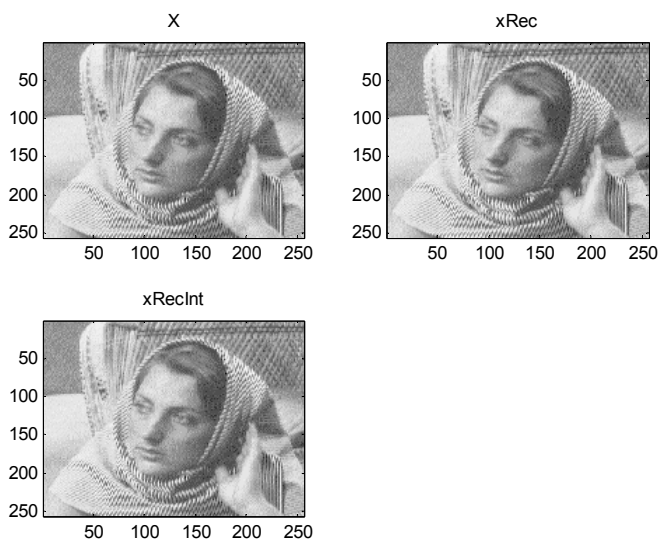


图 8-16 二维提升小波重构

计算结果如下：

```

err =
    0
errInt =
    0

```

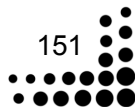
**【例 8-19】**对提升小波分解的某层系数进行单支重构。

程序清单如下：

```

clear;
clc;
% 得到 Haar 小波的提升方案
lshaar=liftwave('haar');
% 添加 ELS 到提升方案中
els={'p',[-0.125 0.125],0};
lsnew=addlift(lshaar,els);

```







```
% 2 层提升小波分解
load woman;
xDec=lwt2(X,lsnew,2);
% 提取近似图像和细节图像
a2=lwtcoef2('a',xDec,lsnew,2,2);
h1=lwtcoef2('h',xDec,lsnew,2,1);
v1=lwtcoef2('v',xDec,lsnew,2,1);
d1=lwtcoef2('d',xDec,lsnew,2,1);
h2=lwtcoef2('h',xDec,lsnew,2,2);
v2=lwtcoef2('v',xDec,lsnew,2,2);
d2=lwtcoef2('d',xDec,lsnew,2,2);
% 检查重构误差
err=max(max(abs(X-a2-h2-v2-d2-h1-v1-d1)))
nbc=size(map,1);
colormap(pink(nbc));
subplot(121);
image(wcodemat(X,nbc));
title('原始图像');
subplot(122);
image(wcodemat(a2,nbc));
title('重构第二层近似图像');
figure;
subplot(231);
image(wcodemat(h1,nbc));
title('第一层水平方向图像');
subplot(232);
image(wcodemat(v1,nbc));
title('第一层垂直方向图像');
subplot(233);
image(wcodemat(d1,nbc));
title('第一层对角方向图像');
subplot(234);
image(wcodemat(h2,nbc));
title('第二层水平方向图像');
subplot(235);
image(wcodemat(v2,nbc));
title('第二层垂直方向图像');
subplot(236);
image(wcodemat(d2,nbc));
title('第二层对角方向图像');
```

程序运行结果如图 8-17 和图 8-18 所示。

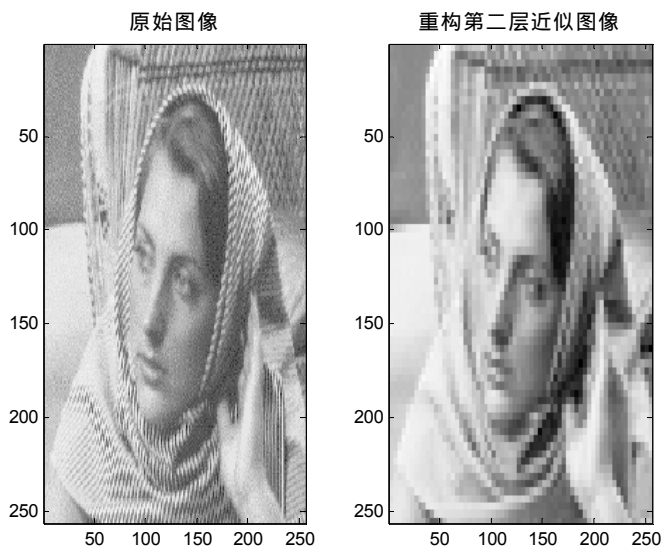


图 8-17 重构低频图像

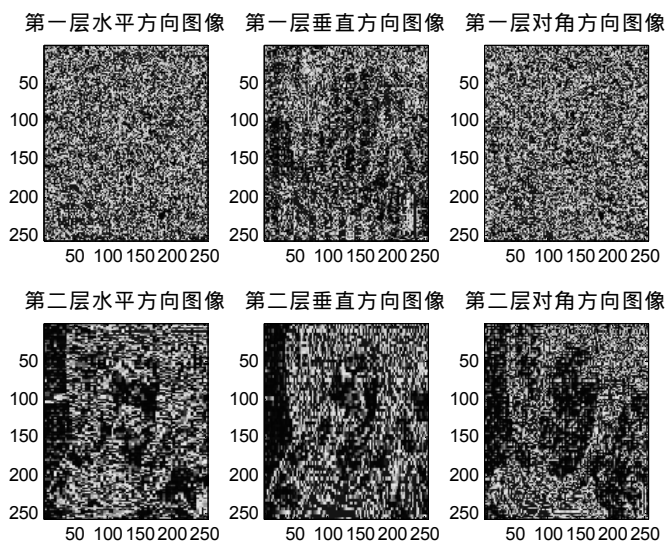


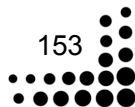
图 8-18 重构高频图像

计算结果如下：

```
err =  
7.1054e-014
```

【例 8-20】ilwt2 函数应用举例。

```
% 使用 Haar 小波，得到相应的提升方案  
lshaar=liftwave('haar');  
% 添加 ELS 到提升方案  
els={'p',[-0.125 0.125],0};
```





```
lsnew=addlift(lshaar,els);  
% 对于简单图像，尺度为 1 进行 LWT  
x=reshape(1:16,4,4);  
[cA,cH,cV,cD]=lwt2(x,lsnew);  
% 对上面的图像，进行整数 LWT  
lshaarInt=liftwave('haar','int2int');  
lsnewInt=addlift(lshaarInt,els);  
[cAint,cHint,cVint,cDint]=lwt2(x,lsnewInt);  
% 进行逆变换  
xRec=ilwt2(cA,cH,cV,cD,lsnew);  
err=max(max(abs(x-xRec)))  
xRecInt=ilwt2(cAint,cHint,cVint,cDint,lsnewInt);  
errInt=max(max(abs(x-xRecInt)))
```

程序运行结果如下：

```
err =  
    0  
errInt =  
    0
```

## 第9章 基于小波变换的回归估计与实现

除了小波在信号与图像方面的广泛应用,在小波分析的应用中还有一类问题的应用也非常广泛,即样本估计。这类问题来源于统计学处理未知样本的思想。在实际应用中,对大量的样本数据,估计其规律是很有实际意义的。而在样本的取值本身,这种规律并不十分明显,导致估计有一定的困难,小波分析提供了一个分离样本之间的关联,找出其内在规律的方法。

### 9.1 密度估计

密度估计是在实际应用中经常遇到的一类问题,我们能获得的信息只有其样本的分布值。可以用数学语言做以下描述:

在密度分布未知的一个样本分布中,设  $(X_i, 1 \leq i \leq n)$  为其样本值,密度分布就是要求出其分布的规律  $\hat{g}$ ,使得在最小偏差的意义上  $\hat{g} = X_i$ 。

众所周知,直方图体现了在一定的可测集上样本的密度分布信息。在19世纪初,法国科学家拉普拉斯通过对直方图进行均衡化的方法,使得我们可以通过简单的函数来描述这种密度分布,这种函数也被称为拉普拉斯-高斯分布。

密度估计的应用很广泛,它也是可信性研究的一个核心方法。比如,可以通过密度估计找出一类电视设备的使用寿命的概率密度分布,或者某一时期的使用性能,或者其他反映物体本质属性的统计特征,如平均损坏时间。另一个非常相似的领域就是医学中的生存期分析,通过密度估计可得到一种医疗方法的剩余生存期。

在已知样本的密度分布信息非常少的情况下,小波对于无参估计是一个很好的工具。我们可以通过小波域的系数,在不必了解样本的统计规律的情况下给出其密度分布函数的形状。

除了波变换外,还有几种其他方法同样能得到样本的密度估计,其中正交基估计同小波变换用于密度估计的思想一致,其他估计方法基于统计窗的方法,包括核心平滑方法等。

基于小波变换的估计方法至少具有和其他方法同样好的性能,在有些情况下会优于其他方法。尤其是在密度分布  $h(x)$  存在奇异点的情况下(如存在间断点),小波方法是一种很好的估计方法。

在用小波变换进行密度估计的过程中,核心思想就是将密度估计问题规定为一个已知模式的回归模型,其具体流程如下:

(1) 通过分层的过程把样本  $X$  转化到  $(X_b, Y_b)$  空间。在分层化过程中,每个  $X_b$  是等间



距划分的, 对于每个层  $j$ ,  $Yb(j)$  为在第  $j$  层中  $X(j)$  的数量。

(2) 将  $Yb$  作为一个普通信号, 进行小波分解。这样隐含的  $Xb$  的定义域就是  $1, 2, \dots, nb$ , 其中,  $nb$  为分层过程确定的层数。

(3) 对分解得到的小波系数作用阈值或掩码 (使用以前降噪或压缩过程中介绍的阈值方法)。

(4) 通过作用阈值后的小波系数重建  $h$  的估计函数  $h1$ 。

(5) 对得到的估计函数  $h1$  做后处理, 将估计函数的定义域通过尺度转换重新转化为  $X$ , 并通过差值把  $h1$  转换为估计函数  $\hat{h}(x)$ 。

在这个流程中, 第 (2) 步到第 (4) 步是标准的基于小波变换的信号处理方法, 而第 (1) 步要依赖于参数  $nb$  (样本的层数)。这里的  $nb$  可以看作度量带宽的参数, 在密度估计的过程中, 由于分层过程是一个平滑的过程,  $nb$  一般比观测长度 (等于  $X$  的长度) 小一些。典型的  $nb$  值一般是  $nb = \text{length}(X)/4$ 。

小波变换进行密度估计的基本原理如下。

设  $x_1, x_2, \dots, x_n$  为  $n$  个独立的同分布的随机变量, 它们的分布服从相同的概率密度函数

$$h = h(x)$$

此处, 概念密度分布函数  $h$  是未知的, 我们只能通过少量的信息对其进行估计。

为了使样本可以进行小波变换, 假设  $h(x) \in L^2(R)$ , 也就是  $\int h^2(x)dx < \infty$ , 这样就可以将  $h(x)$  分解为各个小波基上的系数。

设  $J$  为一个初始化参数, 则  $h(x)$  可以在小波域表示为

$$h = \sum_k \alpha_{J,k} \Phi_{J,k} + \sum_{j=-}^J \sum_k d_{j,k} \Psi_{j,k} = A_J + \sum_{j=-}^J D_j$$

对  $h(x)$  的估计函数  $\hat{h} = \hat{h}(x)$  要用到这些小波分解系数的一部分, 对样本进行估计的基本思想如下。

为了对概率密度函数  $h(x)$  进行估计, 只需要对  $h$  的小波分解系数  $\alpha_{J,k}$  和  $d_{j,k}$  进行估计。

由以前的知识可知, 小波分解系数  $\alpha_{J,k}$  定义为

$$\alpha_{J,k} = \int \Phi_{J,k}(x)h(x)dx$$

类似地, 有

$$d_{j,k} = \int \Psi_{j,k}(x)h(x)dx$$

这可得到一个关于  $\alpha_{J,k}$  的非常有趣的解释。由于  $h(x)$  为概率密度函数, 如果把  $\Phi_{J,k}(x)$  看作一个随机变量,  $\int \Phi_{J,k}(x)h(x)dx$  就代表了该随机变量的数学期望  $E[\Phi_{J,k}(x_i)]$ 。一般情况下, 数学期望是通过简单的求均值得到的, 即

$$\hat{\alpha}_{J,k} = \frac{1}{n} \sum_{i=1}^n \Phi_{J,k}(x_i)$$

同样地, 细节系数  $d_{j,k}$  的数学期望可表示为

$$\hat{d}_{j,k} = \frac{1}{n} \sum_{i=1}^n \Psi_{j,k}(x_i)$$

由于观测集的长度为  $n$ , 是有限的, 那么就可以通过一个小波系数的集合来进行估计,



该集合是在位置  $k$  上的层数为  $0 \sim J$  的小波系数的一个子集。

然后再根据以前在降噪和压缩算法中用到的思想，认为模值的系数对估计的影响不大，将其置为零，这个过程中取阈值为  $t$ 。这样，将处理的系数还原，就得到了对  $h$  的一个估计为

$$\hat{h} = \sum_k \hat{\alpha}_{J,k} \Phi_{J,k} + \sum_{j=-\infty}^J \sum_k \hat{d}_{j,k} l_{\{\|\hat{d}_{j,k}\| > t\}} \Psi_{j,k}$$

式中， $\hat{d}_{j,k} l_{\{\|\hat{d}_{j,k}\| > t\}}$  表示通过作用阈值  $t$  处理后的细节系数。

这种对系数进行阈值处理的好处在于避免了保留所有系数可能引起的波动。

从计算的角度来讲，这种方法在进行快速计算时有一定的问题，因为  $x_i$  并不是等间距划分的。可以通过如下办法来解决这个问题。

引入样本  $x$  的均衡化直方图  $\hat{H}$ ， $\hat{H}$  含有  $nb$  个类，由每个类的中心组成向量  $\mathbf{Xb}$ ，由每个  $x_i$  在该类中出现的频率组成向量  $\mathbf{Yb}$ 。那么对第  $r$  个类，有

$$\hat{H}(x) = \frac{\mathbf{Yb}(r)}{n}$$

这样，可通过  $\hat{H}$  来重新表示上面的公式，即

$$\hat{d}_{j,k} = \frac{1}{n} \sum_{i=1}^n \Psi_{j,k}(x_i) \approx \frac{1}{n} \sum_{r=1}^{nb} \mathbf{Yb}(r) \Psi_{j,k}[\mathbf{Xb}(r)] \approx c \int \Psi_{j,k}(x) \hat{H}(x) dx$$

式中， $c$  表示每个类的长度。

此处使用“ $\approx$ ”是因为在使用直方图  $\hat{H}$  代替  $x_i$  和对每个类的间隔做近似的时候损失了一些信息。

最后一个“ $\approx$ ”为我们提供了一些非常有效的办法。它说明，对细节系数  $d_{j,k}$  的估计  $\hat{d}_{j,k}$  乘上一个常数  $c$  后，就可以表示为直方图  $\hat{H}$  在层数  $j$ 、位置  $k$  的小波分解的细节系数。同样的结果也适用于  $\hat{\alpha}_{J,k}$ 。

最后一个“ $\approx$ ”表明了对概率密度函数的小波分解系数的估计  $\hat{d}_{j,k}$  和  $\hat{\alpha}_{J,k}$  同样可以通过求直方图  $\hat{H}$  的小波分解系数来近似求得。这样，在计算上的障碍就解决了，如果在第  $J$  层的小波系数已知或已经求得，就可通过 mallat 算法快速地求得这些系数值。

这样就可以通过对小波系数的估计  $\hat{d}_{j,k}$  和  $\hat{\alpha}_{J,k}$  来建立对样本的估计  $\hat{h}$ ，其中所使用的技巧同计算直方图类似，也是通过把不规则空间中的  $x$  值转换到均匀划分的空间中来实现的，这个过程称为分类（binning）。

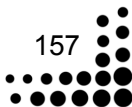
下面介绍利用小波工具箱的一维小波变换的密度估计。

在命令窗口中输入“wavemenu”，打开小波分析工具，实现小波变换的密度估计工具的具体操作步骤如下。

(1) 打开一维小波变换的密度估计。首先，在窗口中输入“wavemenu”，弹出小波工具箱主界面，然后单击右侧第二行 Density Estimation 1-D 按钮。

(2) 下载数据源。选择 File|Load Data for Density Estimation 项，在弹出的对话框中选择 ex1cusp2.mat 文件，它的路径为 toolbox/wavelet/wavedemo，单击 OK 按钮，即可将数据下载到此工具中，如图 9-1 所示。

(3) 执行 3 层小波系数分解。在图 9-1 所示的界面中选择 sym4 小波进行 3 层分解。单击





Decompose 按钮，经过短暂的计算后，工具显示分解图，如图 9-2 所示。

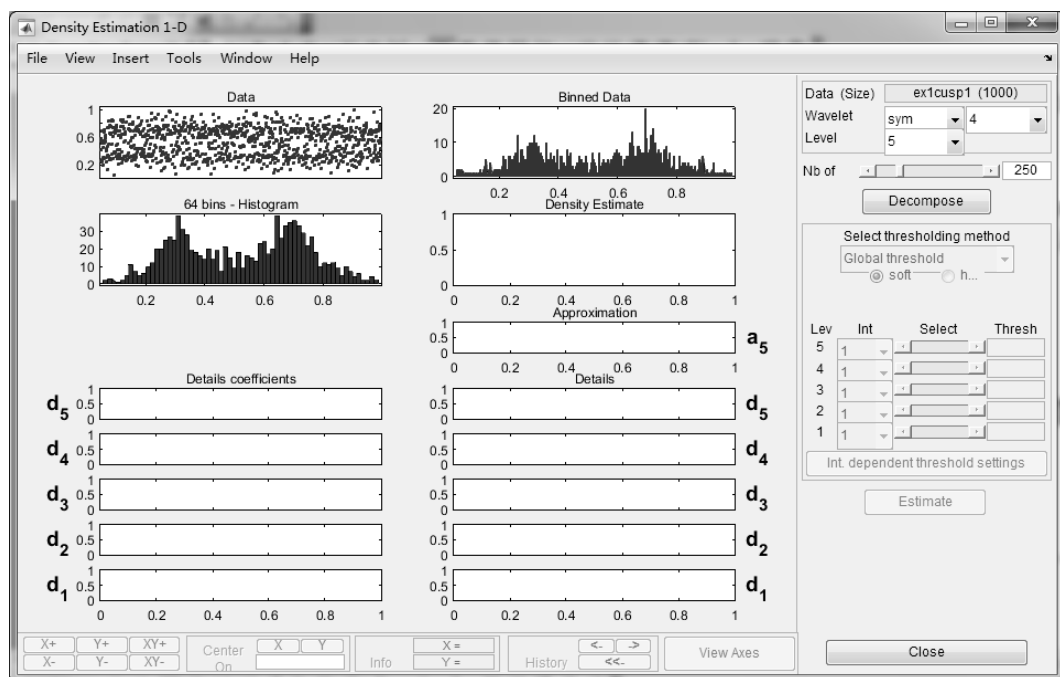


图 9-1 数据源

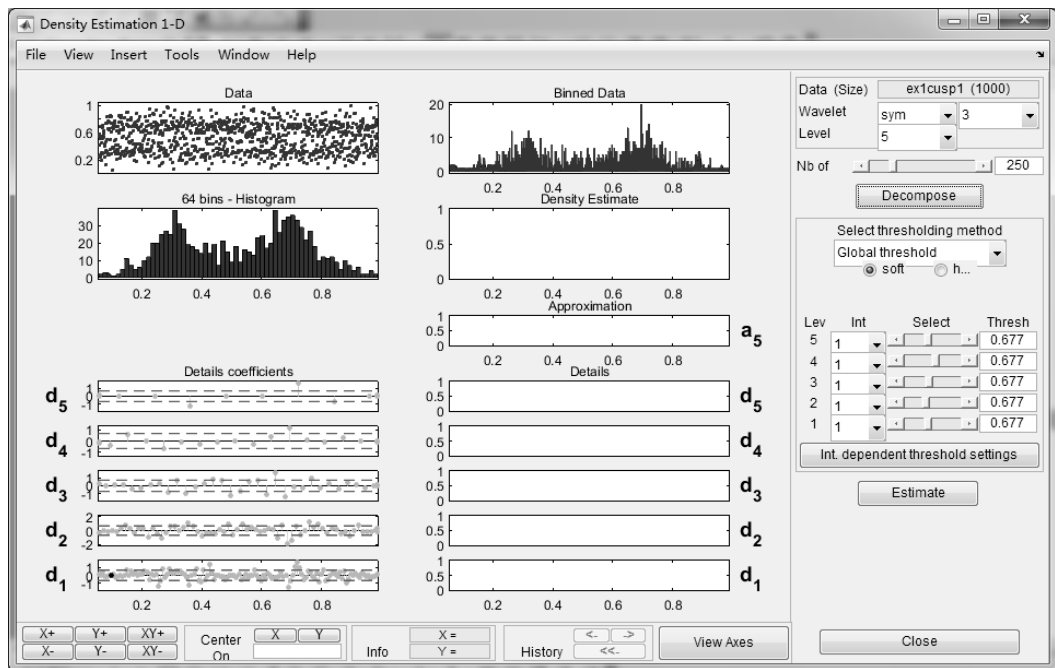


图 9-2 分解图

(4) 选择阈值。在右上方的 Select thresholding method 下选择全局软件阈值参数，单击

Int.dependent threshold settings 按钮，得到如图 9-3 所示的阈值区。

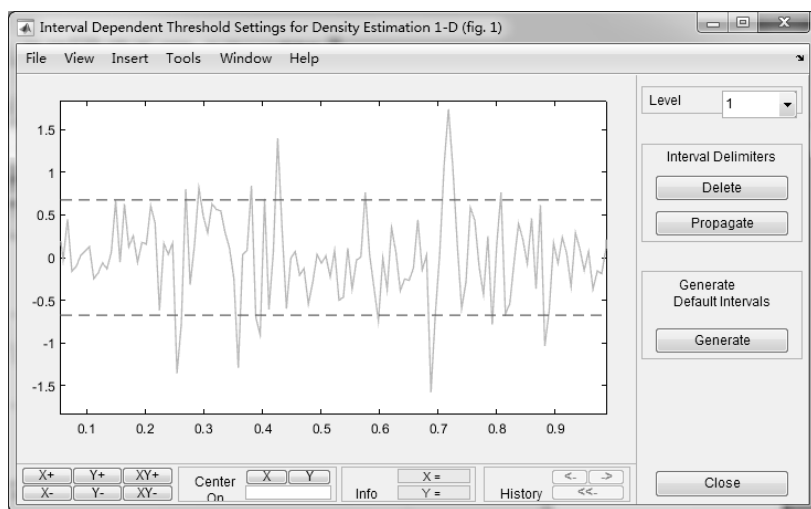


图 9-3 阈值区

(5) 密度估计。单击图 9-2 中的 Estimate 按钮（在 Int.dependent threshold settings 按钮的下面），出现密度估计窗，如图 9-4 所示。可以看出，图 9-4 中的密度曲线很不规则，它是低频信号  $a_4$  和通过阈值处理后的系数重构得到的高频信号的叠加。

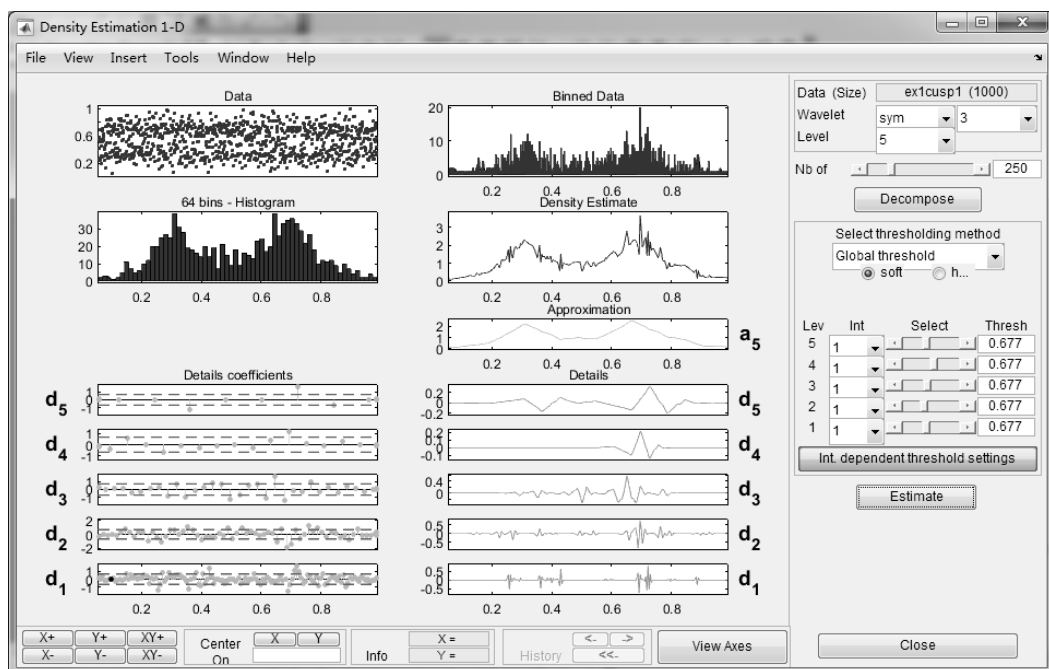


图 9-4 密度估计窗

(6) 保存数据。通过选择图 9-4 所示窗口中的主菜单的 File|Save Density 项（见图 9-5），可保存密度分布，在弹出的对话框中命名为“li7\_1”。



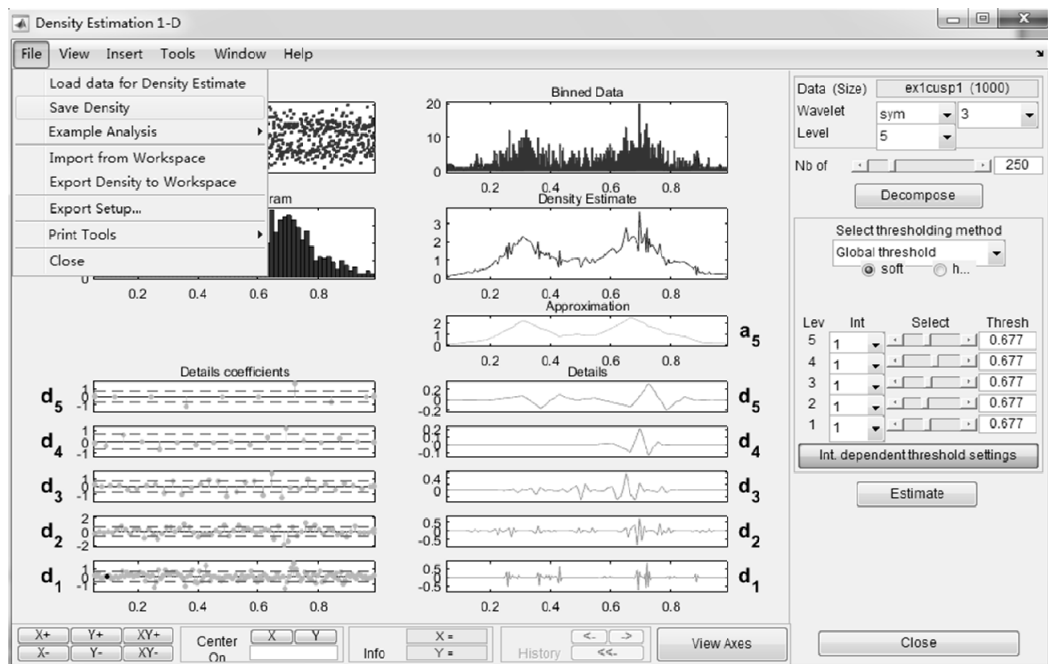


图 9-5 保存密度分布数据

在命令窗口中输入：

```
>> clear
>> load li7_1;
>> whos
```

得到以下结果：

Name	Size	Bytes	Class	Attributes
thrParams	1x5	420	cell	
wname	1x4	8	char	
xdata	1x250	2000	double	
ydata	1x250	2000	double	

密度分布由变量 xdata 和 ydata 给出，它们的长度和上面估计过程中选取的一样。

## 9.2 回归估计

回归估计也是一类在实际应用中非常广泛的问题，它要解决的问题是在一组随机变量  $Y$  和另一组随机变量  $X$  之间建立联系。回归模型给出了一些  $Y$  的变化与  $X$  的变化有关或者影响  $X$  的变化的信息。回归函数  $f$  给出了这种关联的核心信息。

类似于降噪模型，回归估计的基本模型为

$$Y = f(X) + e$$

其中， $e$  表示这种估计的残余部分。



## 9.2.1 回归模型

密度估计是根据一个确定的变量（如时间、空间）来估计另一个随机变量的密度分布。而回归估计不同，要在两组随机变量中找出关联，所以不存在像时间或空间这种理想的、均匀分布的变量空间。因此，在大部分情况下需要假定这种关联的形式，也就是回归模型。

最简单的一种模型是线性模型，服从如下分布：

$$Y = aX + b + e$$

另一种复杂些的模型是建立在一族参数化函数上的，比如：

$$f(X) = \cos(wX)$$

需要讨论的是当  $f$  是完全未知的情况时，回归问题就称为无参估计问题。这种问题可以通过统计窗方法或小波方法来解决。

由于在实际应用中对问题本质能获得的信息少之又少，所以无参的回归估计就成为我们解决问题的强有力的工具，以下是几个典型的通过这种方法解决的问题实例：

- 在金属学中，通过碳纤维的比例解释应力强度；
- 在营销学中，找出经济指标和房屋价格之间的关联；
- 在空气污染研究领域，找出每日最大臭氧密度和每日最高温度之间的关联。

对于这类问题，有两种设计模式，一种是固定的设计模式，另一种是随机的设计模式，两者间的区别就是估计函数  $f$  的状态。

(1) 固定的设计模式。 $X$  的取值范围是设计者预先定义好的，比如一周之中的每一天、产品的使用期或者潮湿度等。通常在这种情况下， $X$  的取值空间是等量划分的。如果  $X$  表示的是时间的取值，那么这类回归估计问题就等同于信号的降噪问题。

(2) 随机的设计模式。 $X$  的取值范围来源于一个测量过程，或者是随机产生的，那么这种设计模式就称为随机的设计模式。在这种模式下，样本的取值空间通常是不规则的，而这种回归预测本身也更具有—般性，因为它研究的是随机变量  $Y$  和随机变量  $X$  之间的关联，也可以说成是随机变量  $Y$  随着随机变量  $X$  的变化的发展。

## 9.2.2 基于小波变换的回归估计

在小波工具箱中，回归估计使用的是固定模式的回归模型，其算法流程如下。

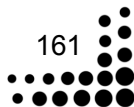
(1) 将样本  $(X, Y)$  规约到  $(X_b, Y_b)$ ，其中， $X_b$  是通过分类过程得到的均匀分布的空间，对于每个类  $i$  都有

$$Y_b(i) = \frac{\sum\{Y(j) | X(j) \in \text{bin}(i), 1 \leq j \leq n\}}{\text{number\_of}\{Y(j) | X(j) \in \text{bin}(i), 1 \leq j \leq n\}}$$

其中， $n$  为样本数量。

为了保证  $Y_b$  的存在，此处约定  $\frac{0}{0} = 0$ 。

(2) 将  $Y_b$  作为信号对其进行小波分解， $X_b$  隐含的定义域是  $1, 2, \dots, n_b$ ，其中， $n_b$  为类的数目。





(3) 对分解得到的小波系数作用阈值或掩码 (使用以前降噪或压缩过程介绍的阈值方法)。

(4) 通过作用阈值后的小波系数重建  $h$  的估计函数  $f_1$ 。

(5) 对得到的估计函数  $f_1$  做后处理, 将估计函数的定义域通过尺度转换重新转化为  $X$ , 并通过差值把  $f_1$  转换为估计函数  $\hat{f}(X)$ 。

与密度估计的流程类似, 第 (2) ~ (4) 步同样是小波降噪或压缩的基本流程, 第 (1) 步的前处理是做了一次平滑, 为了尽量消除随机信号引入的不确定性, 这个过程同样依赖于带宽参数  $nb$ , 也就是分类的数量。由于是平滑过程,  $nb$  要比观察得到的样本数量小。

这个流程的基本原理如下。

回归估计和密度估计的基本思想是一致的, 都是通过对小波系数的估计重建对样本的估计, 它们的主要区别在于选用的模型不同。

二者另一个主要区别就是处理密度的步骤: 回归估计中要建立两个随机变量  $X$  和  $Y$  的概率密度分布, 在密度估计中只需要建立一个。

此处用到的回归模型是线性模型  $Y_i = f(X_i) + \varepsilon_i$ , 其中,  $\varepsilon_i$  是一组独立的同分布的随机变量组成的序列;  $X_i$  是根据未知的密度分布  $h$  随机产生的一组随机序列。

同样, 假定  $(X_1, Y_1), \dots, (X_n, Y_n)$  也是一个独立的同分布的随机变量序列。

下面求出对未知函数  $f$  的回归估计  $\hat{f}$ 。

为了方便表述, 引入函数  $g = fh$ , 则在  $\frac{0}{0} = 0$  的约定下, 有

$$f = \frac{g}{h}$$

这样, 就可通过前面密度估计的方法得到  $h$  的估计  $\hat{h}$ , 再通过  $g$  的估计  $\hat{g}$  就可得到  $f$  的估计

$$\hat{f} = \frac{\hat{g}}{\hat{h}}$$

此处对  $h$  的估计仍然通过均衡化的直方图来求得。

把  $X$  值分为  $nb$  个类, 第 1 个类的中心存在  $Xb(1)$  中, 该类中出现的  $X$  值的频率存放在  $n(1)$  中; 定义  $Yb(1)$  为该类中所有  $X$  值对应的  $Y$  值的总和除以  $X$  值出现的频率。

使用密度估计中的方法即可得到  $f$  的小波系数的估计定义:

$$\hat{\alpha}_{j,k} = \frac{1}{n} \sum_{i=1}^n Y_i \Phi_{j,k}(X_i)$$

$$\hat{d}_{j,k} = \frac{1}{n} \sum_{i=1}^n Y_i \Psi_{j,k}(X_i)$$

类似密度估计中用到的方法, 用对  $Yb$  的小波分解系数来近似上面得到的小波系数估计, 即

$$\hat{\alpha}_{j,k} = \frac{1}{n} \sum_{i=1}^n Yb(1) \Phi_{j,k}[X(1)]$$

$$\hat{d}_{j,k} \approx \frac{1}{n} \sum_{i=1}^{nb} Yb(1) \Psi_{j,k}[Xb(1)]$$



同理，可运用通过小波变换近似求解系数估计的快速算法。

### 9.2.3 小波变换实现回归估计

本小节将介绍利用小波工具箱的一维小波变换的回归估计。

在 MATLAB 命令窗口中输入“wavemenu”命令，弹出小波工具箱主界面。实现一维小波变换的密度估计工具的操作步骤如下。

(1) 打开一维小波变换的密度估计。先在窗口中输入“wavemenu”命令，弹出小波工具箱主界面，然后单击右侧的 Regression Estimation 1-D 按钮。

(2) 下载数据源。选择 File|Load|Data for fixed Design Estimation 项，在弹出的对话框中选择 noisbump.mat，它的路径为 toolbox/wavelet/wavedemo，单击 OK 按钮，这样数据就被下载到工具中，如图 9-6 所示。

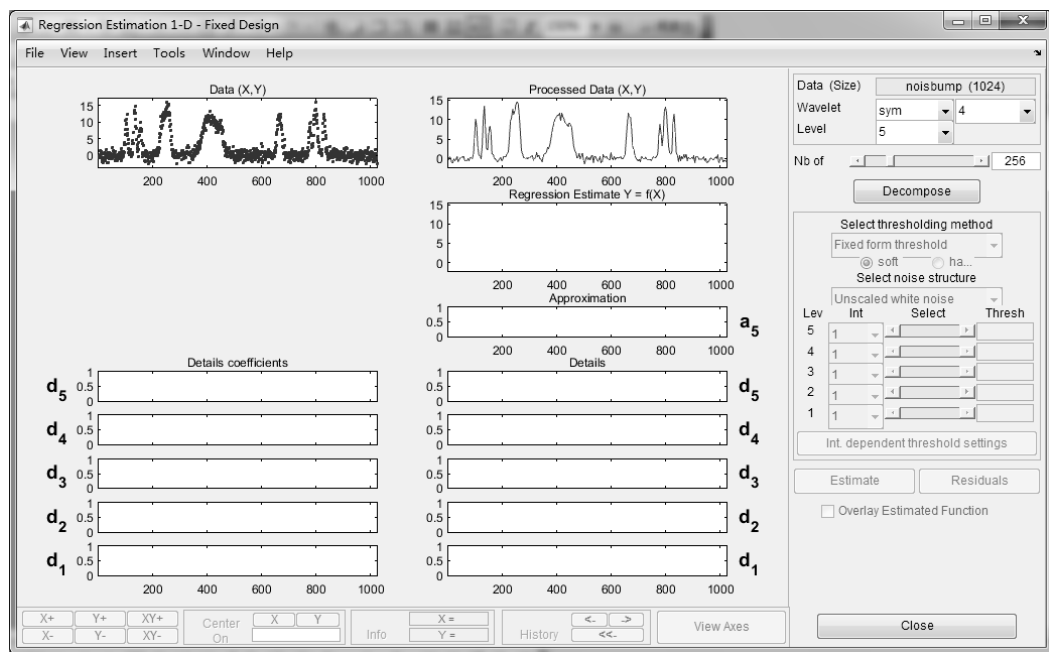
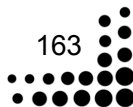


图 9-6 数据源

(3) 执行小波系数分解。在图 9-6 所示的界面中选择 sym4 小波进行 4 层分解。单击 Decompose 按钮，经过短暂的计算后，工具显示分解图，如图 9-7 所示。

(4) 选择阈值。在右上方的 Select thresholding method 下选择全局阈值参数，单击 Int.dependent threshold settings 按钮，得到如图 9-8 所示的阈值区。

(5) 回归估计。单击图 9-7 上的 Estimate 按钮（在 Int.dependent threshold settings 按钮下侧），出现回归估计窗，如图 9-9 所示。可以看出，图 9-9 中的回归估计是位于它下面第一层的由低频信号 a4 和通过阈值处理后的高频信号的叠加。



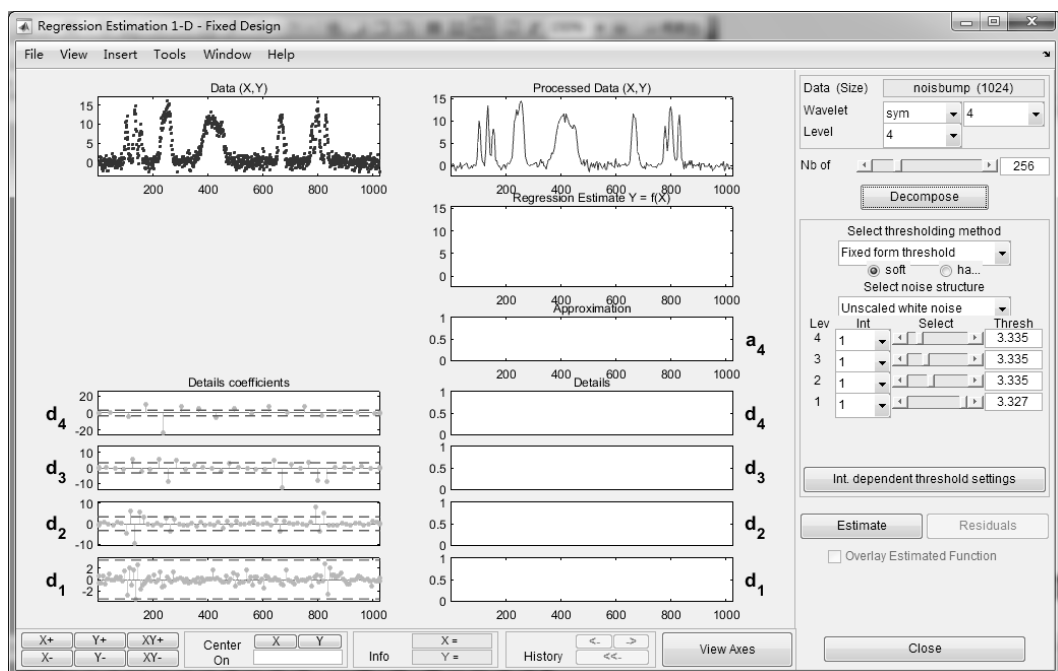


图 9-7 分解图

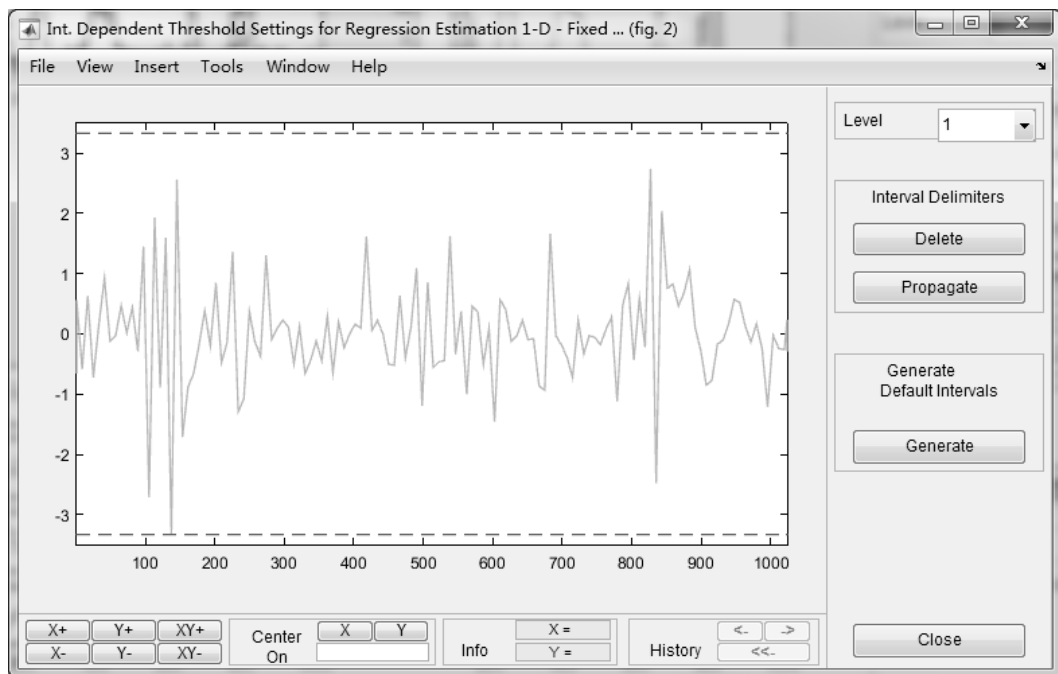


图 9-8 阈值区

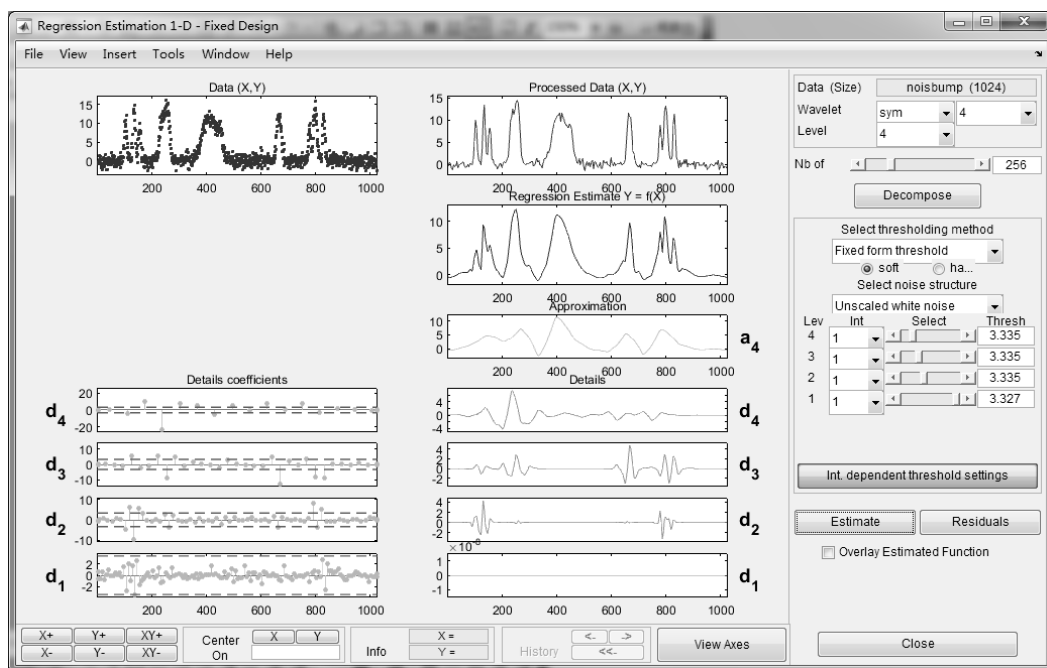


图 9-9 回归估计窗

(6) 显示统计值。单击 Residuals 按钮，弹出回归后的数据与原始数据的误差统计值，如图 9-10 所示。

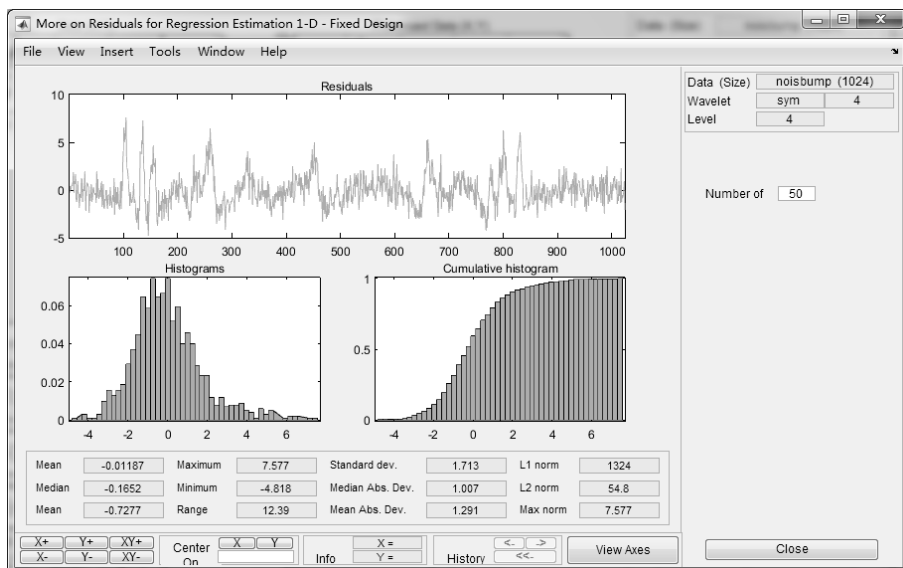
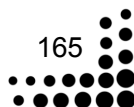


图 9-10 误差统计值

(7) 保存数据。可以通过选择图 9-9 所示窗口中的主菜单的 File|Save Estimation Function 项来保存回归分布，在弹出的对话框中命名为“li7\_2”。





在命令窗口中输入：

```
>> load li7_2;  
>> whos
```

得到以下结果：

Name	Size	Bytes	Class	Attributes
thrParams	1x4	336	cell	
wname	1x4	8	char	
xdata	1x256	2048	double	
ydata	1x256	2048	double	

可以看出，估计函数由 xdata 和 ydata 给出，它们的长度和上面估计过程中选取的一样。

## 第10章 信号的突变点检测

### 算法分析与实现

利用小波多分辨率分析的特性可将突变信号进行多尺度分解，然后通过分解后的信号来确定突变信号的位置。Lipschitz 指数被用来定量描述函数的奇异性。当小波变换尺度越来越精细时，小波变换的模极大值信号在突变点的衰减速度取决于信号在突变点的 Lipschitz 指数。小波变换不仅可以确定突变点发生的时间，而且可以进一步判断突变的性质。

长期以来，傅里叶变换是研究信号奇异性的主要工具，其方法是研究信号在傅里叶变换域的衰减，以推断信号是否具有奇异性及奇异性大小。但傅里叶变换缺乏空间局部性，它只能确定一个信号突变的整体性质，而难以确定突变点在空间的位置及分布情况。由于小波具有空间局部性，它能“聚焦”于信号的局部结构，因此，利用小波变换来确定信号的突变性位置更有效。

### 10.1 信号的突变性与小波变换

S.Mallat 将函数（信号）的局部奇异性与小波变换后的模局部极大值联系起来，通过小波变换后的模极大值在不同尺度上的衰减速度来衡量信号的局部奇异性。

**定理** 设小波  $\psi(t)$  是实函数且连续，具有衰减性  $|\psi(t)| \leq K(1+|t|)^{-2-\varepsilon}$ ， $(\varepsilon > 0)$ ， $f(t) \in L^2(R)$  在区间  $I$  上是一致 Lipschitz 指数  $\alpha$  ( $-\varepsilon < \alpha < 1$ )，则存在常数  $c > 0$ ，使得对  $\forall a, b \in I$ ，其小波变换满足

$$|(Wf)(a, b)| \leq ca^{\alpha + \frac{1}{2}} \quad (10-1)$$

反之，若对于某个  $\alpha$  ( $-\varepsilon < \alpha < 1$ )， $f \in L^2(R)$  的小波变换满足式 (10-1)，则  $f$  在  $I$  上具有一致 Lipschitz 指数  $\alpha$ 。

若  $t_0$  是  $f(t)$  的奇异点，则  $|(Wf)(a, b)|$  在  $b = t_0$  处取得极大值，即此时式 (10-1) 等号成立。

在二进制小波变换情况下，式 (10-1) 变成

$$|(Wf)(2^j, b)| \leq c \times 2^{j(\alpha + \frac{1}{2})} \quad (10-2)$$

在信号和图像处理中，常常使用卷积型小波变换。为此，这里引入卷积型小波变换的概念。





定义 设  $f(t), \psi(t) \in L^2(R)$  , 记

$$\psi_s(t) = \frac{1}{s} \psi\left(\frac{t}{s}\right), \quad s > 0 \quad (10-3)$$

则称

$$(Wf)(s, b) = f * \psi_s(b) = \frac{1}{s} \int_{-\infty}^{+\infty} f(t) \psi\left(\frac{b-t}{s}\right) dt \quad (10-4)$$

为  $f(t)$  的卷积型小波变换, 也称  $f(t)$  的小波变换。

在定理中, 如果将  $f(t)$  的小波变换理解成卷积型小波变换, 则式 (10-1) 和式 (10-2) 就分别变成

$$|(Wf)(s, b)| \leq cs^\alpha \quad (10-5)$$

$$|(Wf)(2^j, b)| \leq c2^{j\alpha} \quad (10-6)$$

式 (10-1) 或式 (10-2) 表明, 若  $\alpha > -\frac{1}{2}$ , 则小波变换模极大值随尺度  $j$  的增大而增大;

若  $\alpha < -\frac{1}{2}$ , 则小波变换模极大值随尺度  $j$  的增大反而减小。这说明, 该信号比不连续信号 (如阶跃信号,  $\alpha = 0$ ) 更加奇异, 这正是噪声对应的情况。上述情况说明, 可以利用小波变换的模极大值随尺度变化的情况来推断信号的突变点类型。

## 10.2 信号的突变点检测原理

信号的突变性检测是先对原信号在不同尺度上进行“磨光”, 再对磨光后信号的一阶或二阶导数检测其极值点或过零点。对信号进行磨光处理, 主要是为消除噪声而不是边缘, 因此磨光函数应是局部化的。常用的磨光函数 (也称平滑函数)  $\theta(t)$  可选取 Gauss 函数或 B-样条函数。磨光函数满足

$$\int_{-\infty}^{+\infty} \theta(t) dt = 1 \quad (10-7)$$

$$\lim_{t \rightarrow \pm\infty} \theta(t) = 0 \quad (10-8)$$

式 (10-7) 表示  $\hat{\theta}(0) = 1$ , 即  $\theta(t)$  可理解为低通滤波器。

由卷积型小波变换的定义及卷积的性质, 有

$$f * \psi_s^{(1)}(t) = f * \left( s \frac{d\theta_s}{dt} \right) = s \frac{d}{dt} [f * \theta_s(t)] \quad (10-9)$$

$$f * \psi_s^{(2)}(t) = f * \left( s^2 \frac{d^2\theta_s}{dt^2} \right) = s^2 \frac{d^2}{dt^2} [f * \theta_s(t)] \quad (10-10)$$

式 (10-9) 和式 (10-10) 中的卷积  $f * \theta_s(t)$  也称磨光 (或平滑) 算子, 表示  $f(t)$  经算子作用后的一个信号。直观的意思是  $f(t)$  的“角点”被磨成光滑弧, 从而使  $f(t)$  变成一个光滑函数  $f * \theta_s(t)$ 。

式 (10-9) 和式 (10-10) 表示  $f(t)$  的小波变换  $(Wf)(s, t) = f * \psi_s^{(1)}(t)$  与  $f * \theta_s(t)$  的一阶导数成正比。而  $(Wf)(s, t) = f * \psi_s^{(2)}(t)$  与  $f * \theta_s(t)$  的二阶导数成正比。这样结合前述定理,



就可以说明小波变换用于突变点检测的基本原理,即选取光滑函数  $\theta(t)$  以后,信号  $f(t)$  的突变点,可以通过检测小波变换  $f * \psi_s^{(1)}(t)$  和  $f * \psi_s^{(2)}(t)$  的模极大值而得到。

利用上述原理,还可以进一步确定突变点的类型。若  $t_0$  点是  $f(t)$  的阶跃突变点,则  $f * \psi_s^{(1)}(t)$  在  $t_0$  处取得非零极大值,从而  $t_0$  是  $\frac{d}{dt}[f * \theta_s(t)]$  的非零极大值点或  $f * \theta_s(t)$  的拐点;若  $t_0$  是  $f(t)$  的局部极值点(或脉冲点),则  $t_0$  是  $\frac{d^2}{dt^2}[f * \theta_s(t)]$  的非零极大值点,即  $\frac{d}{dt}[f * \theta_s(t)]$  的过零点。于是对于给定的尺度  $s$ ,  $f * \psi_s^{(1)}(t)$  的非零极大值点是  $f(t)$  的阶跃突变点,  $f * \psi_s^{(2)}(t)$  的非零极大值点是  $f(t)$  的局部极值点。

对结合  $\theta(t)$  选取 Gauss 函数来说,常用反对称小波  $\psi^{(1)}(t)$  检测阶跃突变点,用对称小波  $\psi^{(2)}(t)$  检测局部极值点。在实际应用中,仅在一个尺度下检测突变点常常很难确定真正的突变点的位置和类型,因此需要多尺度检测。只有在多个尺度上都是极值点的位置才是真正的突变点所在位置。

### 10.3 实验结果与分析

输入: 一维信号  $\{f[n]\}, n=0,1,\dots,N-1$ 。

输出: 突变点位置及类型。

步骤:

(1) 选择小波  $\psi$ , 分解层次  $J$  和阈值  $T$ 。

(2) 对  $\{f[n]\}$  进行二进制小波变换,得各层小波系数  $\{(W_1 f)[k]\}, \{(W_2 f)[k]\}, \dots, \{(W_J f)[k]\}$ 。

(3) 对  $\{(W_j f)[k]\}, j=1,\dots,J$  进行阈值处理,即若  $|(W_j f)[k]| < T$ , 则  $(W_j f)[k] = 0$ 。

(4) 检测  $\{(W_j f)[k]\}, j=1,\dots,J$  的模极大值点,即如果  $k=m$  是极大值点,则满足下面 3 个条件:

$$\begin{aligned} & |(W_j f)[m]| > T; \\ & |(W_j f)[m]| > |(W_j f)[m-1]| \text{ 且 } |(W_j f)[m]| > |(W_j f)[m+1]|; \\ & |(W_j f)[m]| > |(W_j f)[m-1]| \text{ 或 } |(W_j f)[m]| > |(W_j f)[m+1]|. \end{aligned}$$

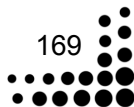
可得  $t_j[0], t_j[1], \dots, t_j[l_j], j=1,2,\dots,J$ 。

(5) 对第(4)步得到的点逐一检查是否是各尺度上的极值点,得到突变点

$$t[0], t[1], \dots, t[s]$$

(6) 根据所选择的小波是反对称的还是对称的,确定得到的点是阶跃边缘点还是局部极值点。

(7) 输出结果。





### 10.3.1 Daubechies 5 小波用于检测突变点

如图 10-1 (a) 所示的原始时域信号是一个含有突变点的信号, 如图 10-1 (b) 所示是利用傅里叶变换对原信号进行处理得到的图像。从图 10-1 (a) 上看, 信号是一条光滑的直线, 但是信号在时间为 500 附近存在突变点, 为了确定阶跃信号的突变点, 采用 Daubechies 5 小波对信号进行处理, 以便确定突变点的位置。

利用傅里叶变换对原始信号进行处理, 可以得到如图 10-1 (b) 所示的图像。从图中可以看出, 信号经过傅里叶变换后能够清楚地确定出原始信号包含的频率值的大小。但是对于确定频率突变点的位置, 傅里叶变换却没有这种能力。

【例 10-1】Daubechies 5 小波用于检测含有突变点的信号。

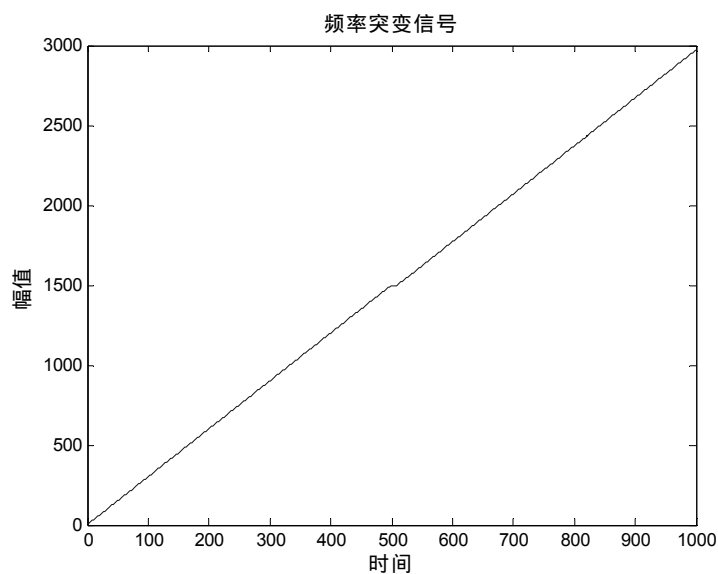
```
clear                                %清除以前的数据
load nearbrk;                        %载入原始信号的波形数据
whos;                                %显示数据的基本信号
figure(1);
plot(nearbrk)
xlabel('时间');ylabel('幅值');        %自定义坐标轴
title('频率突变信号');              %自定义坐标
figure(2);
f=fft(nearbrk);                      %对信号进行傅里叶变换
plot(abs(f));                        %显示处理后的信号图像
title('傅里叶变换后的信号示意图')   %自定义标题
figure(3);
[d,a]=wavedec(nearbrk,3,'db5');      %对原始信号进行 3 层小波分解
a3=wrcoef('a',d,a,'db5',3);          %重构 3 层近似系数
d3=wrcoef('d',d,a,'db5',3);          %重构 1~3 层细节系数
d2=wrcoef('d',d,a,'db5',2);
d1=wrcoef('d',d,a,'db5',1);
subplot(411);plot(a3);ylabel('近似信号 a3'); %显示各层小波系数
title('小波分解后示意图');
subplot(412);plot(d3);ylabel('细节信号 d3');
subplot(413);plot(d2);ylabel('细节信号 d2');
subplot(414);plot(d1);ylabel('细节信号 d1');
xlabel('时间');
```

程序运行结果如图 10-1 所示。

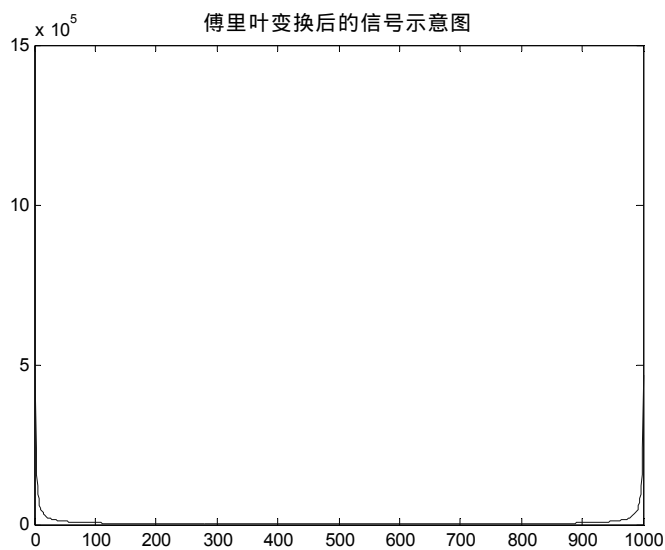
利用小波变换对原始信号进行处理, 可以得到如图 10-1 (c) 所示的小波分解示意图。从图中可见, Daubechies 5 小波分解后的 3 层高频重构图形可清楚地确定突变点的位置, 而傅里叶变换却没有这种能力。



从图 10-1 (c) 中同样可以看出, 第 1 层分解的 d1 高频系数重构的图像比 d2、d3 高频系数重构的图像更清楚地确定了信号的突变点的位置。

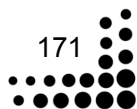


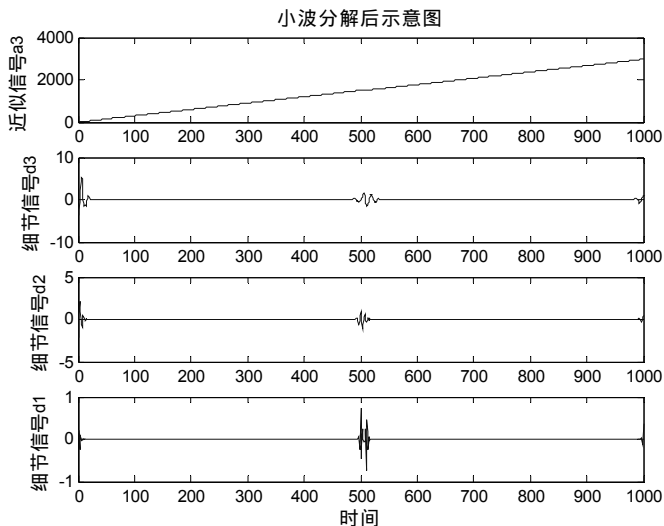
(a) 原始信号



(b) 傅里叶变换后的信号

图 10-1 检测含有突变点的信号



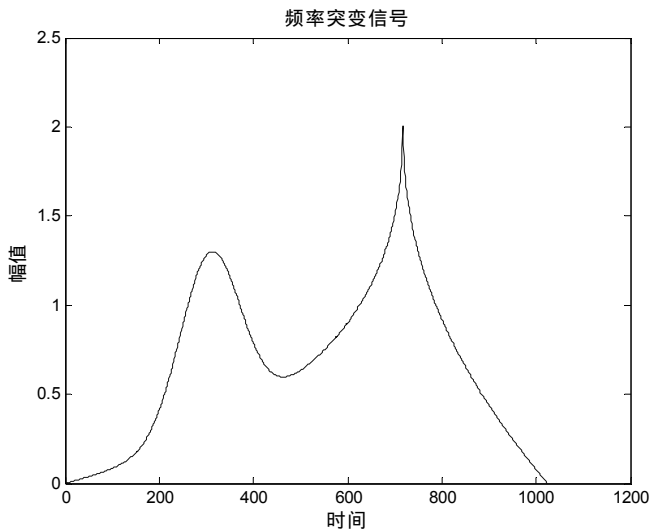


(c) 小波变换后的示意图

图 10-1 检测含有突变点的信号 (续)

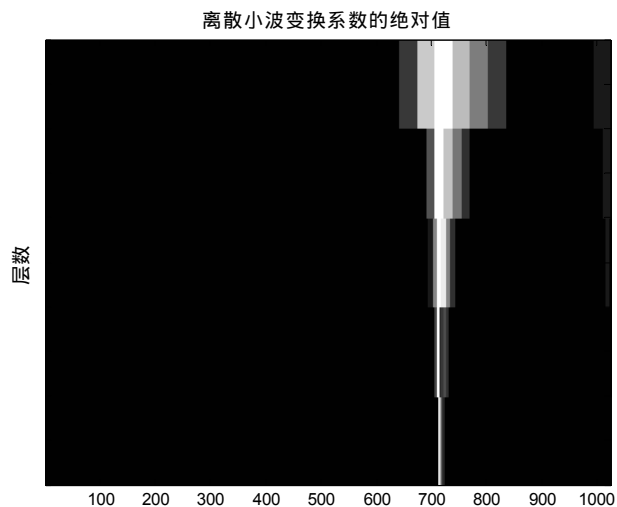
### 10.3.2 Daubechies 6 小波用于检测突变点

如图 10-2 (a) 所示的原始信号是含有突变点的信号。为了确定该突变点的时间, 采用 Daubechies 6 小波进行连续变换后, 再对系数进行分析处理, 以便确定突变点所在的时间。对原始信号用 Daubechies 6 进行 5 层小波分解, 分解层数 1~5 对应的尺度分别为 2、4、8、16 和 32。相应系数绝对值的图像如图 10-2 (b) 所示。

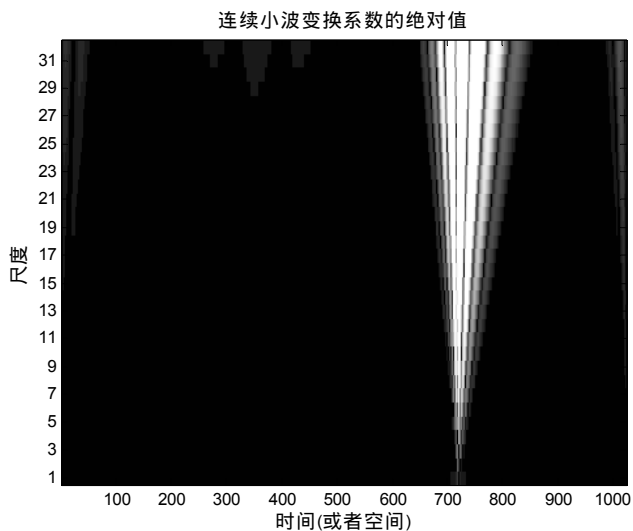


(a) 原始信号

图 10-2 小波用于检测突变点



(b) Daubechies 6 离散小波变换系数



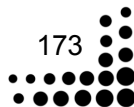
(c) Daubechies 6 连续小波变换系数

图 10-2 小波用于检测突变点 (续)

对原始信号使用 Daubechies 6 小波在尺度 1 ~ 32 上进行连续小波变换。相应系数绝对值的图像如图 10-2 (c) 所示。

**【例 10-2】** Daubechies 6 小波用于检测突变点。

```
clear                                %清除以前的数据
load cuspamax;                       %载入原始信号的波形数据
whos;                                %显示数据的基本信号
figure(1)
plot(cuspamax)
```





```
xlabel('时间');ylabel('幅值');          %自定义坐标轴
title('频率突变信号');                %自定义坐标
figure(2)
[c,l]=wavedec(cuspamax,5,'db6');
cfd=zeros(5,1024);
for k=1:5
    d=detcoef(c,l,k);
    d=d(ones(1,2^k),:);
    cfd(k,:)=wkeep(d(:),1024)
end
cfd=cfd(:);
I=find(abs(cfd)<sqrt(eps));
cfd(I)=zeros(size(I));
cfd=reshape(cfd,5,1024);
colormap(pink(64));
img=image(flipud(wcodemat(cfd,64,'row')));
set(get(img,'parent'),'YtickLabel',[]);
title('离散小波变换系数的绝对值')
ylabel('层数')

figure(3)
ccfs=cwt(cuspamax,1:32,'db6','plot');
title('连续小波变换系数的绝对值')
colormap(pink(64));
ylabel('尺度')
xlabel('时间(或者空间)')
```

程序运行结果如图 10-2 所示。

从图 10-2 (c) 的原始信号连续小波变换系数示意图可以清楚地看出，在时间为 710 时，小波系数出现了一个倒锥形的区域，因此可以推断在该区域存在突变点。本实验再次说明，小波分析在检测信号突变点（奇异点）应用中具有傅里叶变换无法比拟的优越性。

小波变换优越于傅里叶变换的地方在于，小波变换能够同时在时域和频域突出信号的局部特性。几乎所有的信号都能够根据从原始数据中提取出来的某些特征来表现信号。小波变换用于信号的突变点的检测，无论采用小波变换系数的模极大点还是过零点方法，都应在多尺度上做综合分析和判断，才能够准确地确定突变点的位置。通常，较小尺度下的小波变换能够减小频率混叠现象，判断突变点位置的准确度较高。

# 第 11 章 图像边缘检测算法 分析与实现

通过检测二维小波变换的模极大点可以确定图像的边缘点。由于小波变换在各尺度上都提供了图像的边缘信息，所以称为多尺度边缘。沿着边界方向将任意尺度下的边缘连接起来可形成该尺度下沿着边界的模极大曲线。小波变换能够把图像分解成多种尺度成分，并对大小不同的尺度成分采用相应的时域或空域取样步长，从而能够不断地聚焦到对象的任意微小细节。小波变换具有的多尺度特性，正好可以用于图像的边缘检测。

小波分析是近 20 多年来发展起来的新兴学科，作为一种快速高效、高精度的近似方法，它是傅里叶分析的一个突破性发展，给许多相关学科的研究领域带来了新的思想，为工程应用提供了一种新的分析工具。边缘检测是图像处理中的重要内容。边缘是图像的最基本特征。目前边缘检测已成为机器视觉领域最活跃的课题之一，在工程应用中占有十分重要的地位。

## 11.1 多尺度边缘检测

通常，沿边缘走向的幅度变化平缓，垂直于边缘走向的幅度变化剧烈。此外，因物体大小不一，它们的边缘也有不同的尺度。边缘点的 Lipschitz 正则性取决于尺度细化过程中模极大值的衰减速度。

在二维情况下，边缘检测算法通过计算图像信号  $f(x, y)$  的梯度矢量

$$\nabla(f) = \left( \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right)$$

的模的局部极大值来寻找图像边缘的空间位置。梯度矢量的方向指出了图像灰度值变化最快的方向。

为了计算图像信号的两个偏导数，需要两个有方向性的二维小波，它们分别是二维平滑函数  $\theta(x, y)$  的偏导数

$$\psi^x(x, y) = -\frac{\partial \theta(x, y)}{\partial x} \text{ 和 } \psi^y(x, y) = -\frac{\partial \theta(x, y)}{\partial y}$$

$\theta(x, y)$  在  $x-y$  平面的积分为 1，且很快地收敛到零。

令

$$\psi_j^x(x, y) = 2^{-j} \psi^x(2^{-j} x, 2^{-j} y), \psi_j^y(x, y) = 2^{-j} \psi^y(2^{-j} x, 2^{-j} y)$$





并定义小波变换的两个分量

$$W^x f(2^j, x, y) = [f(u, v), \psi_j^x(u - x, v - y)] = f * \overline{\psi_j^x}(x, y)$$

$$W^y f(2^j, x, y) = [f(u, v), \psi_j^y(u - x, v - y)] = f * \overline{\psi_j^y}(x, y)$$

其中,  $\overline{\psi_j^x}(x, y) = \psi_j^x(-x, -y)$ ,  $\overline{\psi_j^y}(x, y) = \psi_j^y(-x, -y)$ 。

任意  $f \in L^2(R^2)$  的二进制小波变换定义为如下函数族:

$$Wf(2^j, x, y) = \{W^x f(2^j, x, y), W^y f(2^j, x, y)\}_{j \in \mathbb{Z}}$$

为确保二进制小波变换的完备性和稳定性, 必须满足如下充分必要条件: 存在两个正常数  $A$  和  $B$ , 对  $\forall(\omega_x, \omega_y) \in R^2 - \{(0, 0)\}$  使

$$A \sum_{j=-\infty}^{\infty} |\hat{\psi}^x(2^j \omega_x, 2^j \omega_y)|^2 + |\hat{\psi}^y(2^j \omega_x, 2^j \omega_y)|^2 \leq B$$

其中,  $\hat{\psi}^x$  和  $\hat{\psi}^y$  分别表示  $\psi^x$  和  $\psi^y$  的二维傅里叶变换。满足上式的  $\{\psi^x, \psi^y\}$  称为二进小波。

对二进小波, 存在重构小波  $\{\tilde{\psi}^x, \tilde{\psi}^y\}$ , 它们的傅里叶变换满足

$$\sum_{j=-\infty}^{\infty} 2^{-2j} [\hat{\tilde{\psi}}^x(2^j \omega_x, 2^j \omega_y) \hat{\psi}^{x*}(2^j \omega_x, 2^j \omega_y) + \hat{\tilde{\psi}}^y(2^j \omega_x, 2^j \omega_y) \hat{\psi}^{y*}(2^j \omega_x, 2^j \omega_y)] = 1$$

因而有

$$f(x, y) = \sum_{j=-\infty}^{\infty} 2^{-2j} [W^x f(2^j, x, y) * \tilde{\psi}_j^x(x, y) + W^y f(2^j, x, y) * \tilde{\psi}_j^y(x, y)]$$

由于  $\{\psi^x, \psi^y\}$  是平滑函数  $\theta(x, y)$  的一阶偏导数, 所以二维二进小波变换的两个分量等价于信号  $f(x, y)$  被平滑后的梯度矢量的两个分量, 即

$$\begin{pmatrix} W^x f(2^j, x, y) \\ W^y f(2^j, x, y) \end{pmatrix} = 2^j \begin{pmatrix} \frac{\partial}{\partial x} (f * \bar{\theta}_j)(x, y) \\ \frac{\partial}{\partial y} (f * \bar{\theta}_j)(x, y) \end{pmatrix} = 2^j \nabla (f * \bar{\theta}_j)(x, y)$$

梯度矢量  $\nabla (f * \bar{\theta}_j)(x, y)$  的模正比于

$$Mf(2^j, x, y) = \sqrt{|W^x f(2^j, x, y)|^2 + |W^y f(2^j, x, y)|^2}$$

而梯度矢量与水平方向的夹角为

$$Af(2^j, x, y) = \begin{cases} \alpha(x, y), & W^x f(2^j, x, y) > 0 \\ \pi - \alpha(x, y), & W^x f(2^j, x, y) < 0 \end{cases}$$

其中,  $\alpha(x, y) = \arctan \left[ \frac{W^y f(2^j, x, y)}{W^x f(2^j, x, y)} \right]$ 。

用二进小波变换实现多尺度边缘检测就是寻找  $Mf(2^j, x, y)$  的局部极大值,  $Af(2^j, x, y)$  指明了边缘的方向。除了边缘的位置和方向外, 还可以用小波变换的衰减速度判断边缘的奇异性。对 Lipschitz 指数  $0 < \alpha < 1$ , 若存在常数  $A > 0$ , 对所有的  $(x, y) \in R^2$ , 使得

$$|f(x, y) - f(x_0, y_0)| \leq A(|x - x_0|^2 + |y - y_0|^2)^{\alpha/2}$$

则称函数  $f$  在  $(x_0, y_0)$  点 Lipschitz  $\alpha$ 。若对区域  $(x_0, y_0) \in \Omega$  内的所有点, 都存在  $A > 0$ ,



使得上式成立，则称函数  $f$  在  $\Omega$  内一致 Lipschitz  $\alpha$ 。可以证明：当且仅当存在  $A > 0$ ，对于所有的  $2^j$  尺度及区域  $\Omega$  内的所有点，使得

$$|Mf(2^j, x, y)| \leq A \cdot 2^{j(\alpha+1)}$$

则  $f$  在  $\Omega$  内一致 Lipschitz  $\alpha$ 。

## 11.2 快速多尺度边缘检测算法

作边缘检测的二维二进小波可以设计为一维二进小波的可分积，具体地说，它们的傅里叶变换为

$$\hat{\psi}^x(\omega_x, \omega_y) = G(\omega_x/2) \hat{\phi}(\omega_x/2) \hat{\phi}(\omega_y/2)$$

$$\hat{\psi}^y(\omega_x, \omega_y) = G(\omega_y/2) \hat{\phi}(\omega_x/2) \hat{\phi}(\omega_y/2)$$

其中， $\hat{\phi}(\omega)$  是一个低通滤波器，而

$$G(x) = -i\sqrt{2}e^{-i\omega/2} \sin(\omega/2)$$

是一个高通数字滤波器。

为了能用滤波器快速实现二维离散二进小波变换，假定尺度函数满足如下二尺度方程：

$$\hat{\phi}(\omega) = \prod_{p=1}^{+\infty} \frac{H(2^{-p}\omega)}{\sqrt{2}} = \frac{1}{2} H\left(\frac{\omega}{2}\right) \hat{\phi}\left(\frac{\omega}{2}\right)$$

若选择尺度函数为  $m$  次样条，即

$$\hat{\phi}(\omega) = e^{-i\omega/2} \left[ \frac{\sin(\omega/2)}{\omega/2} \right]^{m+1}, \quad \varepsilon = \begin{cases} 0, & m \text{ 为奇数} \\ 1, & m \text{ 为偶数} \end{cases}$$

则可得

$$H(\omega) = \sqrt{2}e^{-i\omega/2} [\cos(\omega/2)]^{m+1}$$

对二进小波变换在所有尺度时都均匀采样，假定采样间隔等于 1，则离散小波系数为

$$d_j^x(n, m) = W^x(2^j, n, m), \quad d_j^y(n, m) = W^y(2^j, n, m)$$

同样，定义原始图像信号为

$$a_0(n, m) = \langle f(x, y), \phi(x-n)\phi(y-m) \rangle$$

$j = 0$  时的平滑图像信号为

$$a_j(n, m) = \langle f(x, y), \phi_j(x-n)\phi_j(y-m) \rangle$$

那么，二维离散二进小波变换的  $a$  trous 算法表示为如下离散卷积形式：

$$a_{j+1}(n, m) = a_j * \overline{h_j h_j}(n, m)$$

$$d_{j+1}^x(n, m) = a_j * \overline{g_j \delta}(n, m)$$

$$d_{j+1}^y(n, m) = a_j * \overline{\delta g_j}(n, m)$$

其中，

$$\overline{h_j h_j}(n, m) = \overline{h_j}(n) \overline{h_j}(m)$$





$$\overline{g_j} \delta(n, m) = \overline{g_j(n)} \delta(m)$$

$$\delta \overline{g_j}(n, m) = \delta(n) \overline{g_j(m)}$$

也就是说,  $a_{j+1}$  是  $a_j$  沿横向和纵向低通滤波的结果, 而  $d_{j+1}^x$  是  $a_j$  沿横向高通滤波的结果,  $d_{j+1}^y$  是  $a_j$  沿纵向高通滤波的结果。

## 11.3 实验结果与分析

设灰度图像  $x(n, m)$  为  $N_x \times M_x$  矩阵, 二维数字滤波器  $h(n, m)$  为  $N_h \times M_h$  矩阵, 输出  $y(n, m)$  为  $(N_x + N_h - 1) \times (M_x + M_h - 1)$  矩阵。仿真程序实现灰度图像  $x(n, m)$  和二维数字滤波器  $h(x, y)$  的离散卷积, 得到输出图像  $y(n, m)$  与输入图像  $x(n, m)$  具有相同的大小。

灰度图像经过数字滤波器等计算后, 矩阵元素可能出现负值。例如, 经过高通滤波, 一般都会出现负值, 因为高通滤波体现了灰度的变化, 如灰度由暗变亮时出现正值, 由亮变暗时就会出现负值, 反之亦然。在进行图像边缘检测时, 主要关心的是小波变换的模, 所以这个仿真程序也主要是计算小波变换的模。程序清单如下:

```
clear all;load chess;
I=ind2gray(X,map);
I=imadjust(I,stretchlim(I),[0 1]);
[N,M]=size(I);
figure
imshow(I);title('(a) 原始图像');
% 设置样条滤波器系数
h=[0.125,0.375,0.375,0.125];
g=[0.5,-0.5];
delta=[1,0,0];
% 在设置分解级数时,逼近 x、y 方向的二进小波系数及梯度绝对值数组清零
J=2;
a(1:N,1:M,1:J+1)=0;
dx(1:N,1:M,1:J+1)=0;
dy(1:N,1:M,1:J+1)=0;
d(1:N,1:M,1:J+1)=0;
% 第 1 级分解,显示第 1 级分解的边缘
a(:, :, 1)=conv2(h,h,I,'same');
dx(:, :, 1)=conv2(delta,g,I,'same');
dy(:, :, 1)=conv2(g,delta,I,'same');
x=dx(:, :, 1);
y=dy(:, :, 1);
d(:, :, 1)=sqrt(x.^2+y.^2);
I=imadjust(d(:, :, 1),stretchlim(d(:, :, 1)),[0 1]);
```



```

figure;
imshow(I);title('(b) 第 1 级小波变换边缘检测');
% 第 2 至 J+1 级分解
lh=length(h);
lg=length(g);
for j=1:J
    lhj=2^j*(lh-1)+1;
    lgj=2^j*(lg-1)+1;
    hj(1:lhj)=0;
    gj(1:lgj)=0;
    for n=1:lh
        hj(2^j*(n-1)+1)=h(n);
    end
    for n=1:lg
        gj(2^j*(n-1)+1)=g(n);
    end
    a(:,j+1)=conv2(hj,hj,a(:,j),'same');
    dx(:,j+1)=conv2(delta,gj,a(:,j),'same');
    dy(:,j+1)=conv2(gj,delta,a(:,j),'same');
    x=dx(:,j+1);
    y=dy(:,j+1);
    d(:,j+1)=sqrt(x.^2+y.^2);
    I=imadjust(d(:,j+1),stretchlim(d(:,j+1)),[0 1]);
    figure(j+2);
    if j==1
        ch='c'
    else
        ch='d'
    end
    imshow(I);title(['( ',ch,') 第',num2str(j+1),'级小波变换边缘检测']);
end

```

仿真程序运行结果如图 11-1 所示。图 11-1 (a) 所示为原始图像, 图 11-1 (b) ~ 图 11-1 (d) 所示依次为第 1 ~ 3 级小波变换的模, 第 1 级小波变换模显示出图像的边缘和纹理, 第 2、3 级小波变换模则主要显示出图像的边缘, 平滑掉了图像细致的纹理结构。

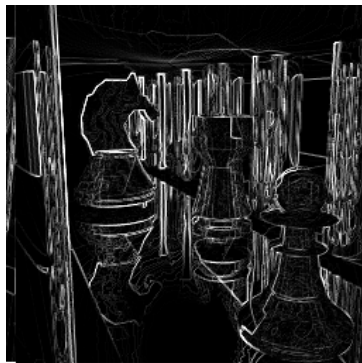
物体的边缘表现为图像局部特征的不连续性, 如灰度值的突变、颜色的突变。边缘常常意味着一个区域的终结和另一个区域的开始。图像边缘有幅度和方向两个特征。这里的创新点就是结合基于灰度图像和边缘图像的二进小波检测方法的优点, 提出了一种基于小波变换的快速检测算法, 它既克服了直接从灰度图像中提取所带来的算法复杂、耗时长缺点, 又克服了一般的边缘提取算法所带来的噪声敏感问题, 从而使误判率降低。



(a) 原始图像



(b) 第1级小波变换边缘检测



(c) 第2级小波变换边缘检测



(d) 第3级小波变换边缘检测



图 11-1 二进小波变换图像边缘检测

## 第 12 章 二维小波变换的算法分析与实现

小波分析的应用是与小波分析理论研究紧密地结合在一起的。现在，它已经在信息产业领域取得了令人瞩目的成就。信号处理已经成为当代科学技术工作的重要部分，信号处理的目的是：准确地分析、诊断，编码压缩和量化，快速传递或存储，精确地重构（或恢复）。从数学的角度来看，信号与图像处理可以统一看作信号处理（图像可以看作二维信号），在小波分析许多应用中，都可以归结为信号处理问题。现在，对于性质随时间稳定不变的信号，处理的理想工具仍然是傅里叶分析。但是实际应用中的绝大多数信号是非稳定的，而特别适用于非稳定信号的工具就是小波分析。

### 12.1 MATLAB 的图像处理

由于图像操作很多，这里仅仅以 MATLAB 窗口的操作、图像的噪声消除和边缘检测为例，来说明该工具的基本使用方法。

#### 12.1.1 MATLAB 图像处理应用举例

（1）MATLAB 的打开。运行 MATLAB 程序，若在桌面上建立了 MATLAB 的快捷方式，双击桌面的 MATLAB 图标即可启动 MATLAB，其命令行窗口界面如图 12-1 所示。

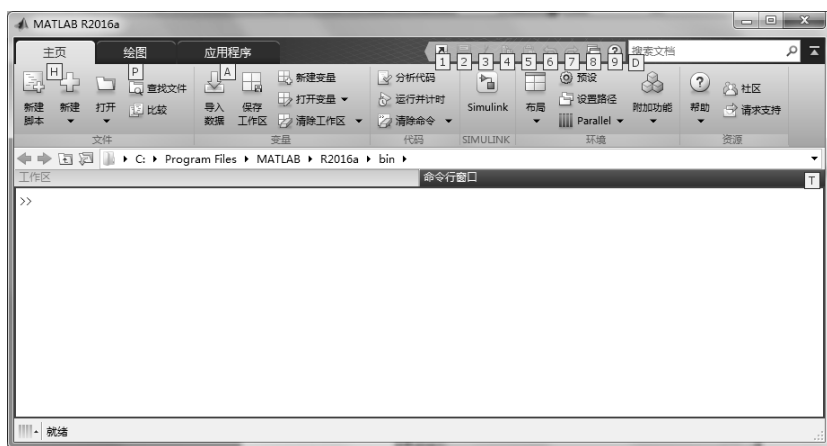


图 12-1 MATLAB 的 Command Window 窗口



(2) 图像输入到计算机。将所需要处理的图像通过数码照相机、U 盘等输入设备输入到计算机中,并确定图像在计算机中存放的位置。例如,图像存在 D:\MATLAB 中。

(3) 打开编辑窗口编写程序。启动 MATLAB 的 M 文件编辑器,在工具栏中选择“新建 脚本”命令,即可建立 M 文件。在编辑窗口中输入要处理图像的源程序,如图 12-2 所示。

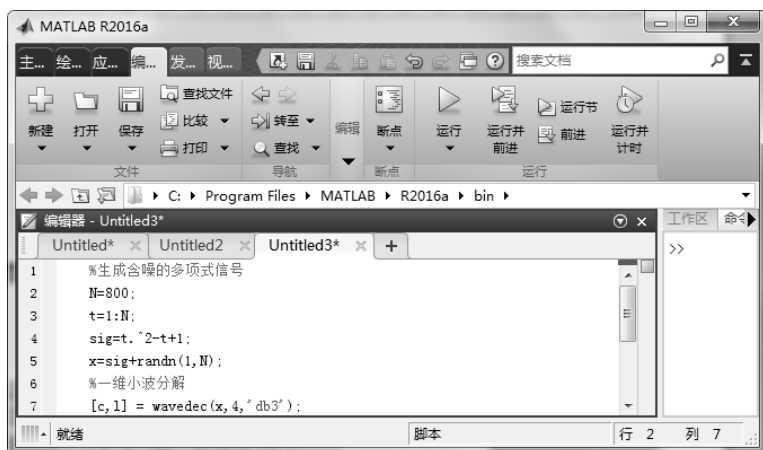


图 12-2 编辑程序

(4) 保存并运行。选择运行,即可运行程序并保存该程序。程序运行结果如图 12-3 所示。

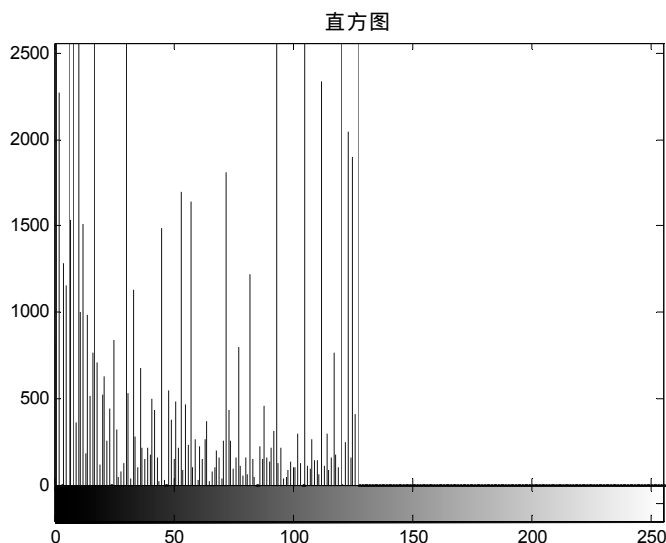


图 12-3 程序运行结果

(5) 保存运行结果。若想将运行结果保存成图片的格式,则需在程序中加入图像 I/O 文件中的图像写入图形文件函数,即 `imwrite(A, filename, fmt)`。若想让图像 I 保存在 D 盘 TAB 文件夹中,文件名为 ex123,文件格式为 BMP,则可用语句“`imwrite(I, 'D:\TAB\ex123.BMP')`”实现。

12.1.2 图像处理基本操作

1. 读取图像并显示

在读取图像之前，首先应清除 MATLAB 所有的工作平台变量，并关闭打开的图形窗口。为此，可使用以下命令行：

```
Clear;
close all;
```

然后使用图像选取函数 `imread`，就可以读取一幅图像。假设 12.1.1 小节读取的图像为 `trees.tif`（该图像是图像处理工具箱自带的），要将它存储在一个名为 `I` 的数组中，可以使用命令：

```
I=imread('trees.tif');
```

然后可以调用 `imshow` 命令来显示图像，即

```
imshow(I)
```

得到显示结果如图 12-4 所示。

2. 检测内存中的图像

使用如下命令可以查看图像 `I` 的存储方式：

```
whos
```

运行后的输出结果如下：

Name	Size	Bytes	Class
I	258x350	90300	uint8 array
Grand total is 90300 elements using 90300 bytes			

以上输出结果说明图像采用 8 位存储方式，并占用了 90300 B 的存储空间。

3. 实现直方图均衡化

由图 12-4 可知，`trees.tif` 对比度比较低，为了更好地观察图像的灰度分布信息，可以用 `imhist` 函数创建描述图像灰度分布的直方图，并使用 `figure` 命令将直方图显示在一个新的图像窗口，如图 12-3 所示。语句如下：

```
figure, imhist(I) %在新图中显示图像 I 的直方图
```

从图 12-3 中可以看出，由于图像的灰度范围比较狭窄，没有覆盖整个灰度范围`[0, 255]`（图像的默认存储类型为 `uint8`），并且图像中灰度值的高低区分不明显，因而不能产生较好的对比效果。要产生较好的对比效果，可以调用 `histeq` 函数将图像的灰度值扩展到整个灰度范围中，从而将数组 `I` 的对比度提高。修改过的图像数据将保存在变量 `I2` 中，并在一个新的图像窗口中显示均衡处理的图像 `I2`。利用以下命令行，可得到如图 12-5 所示的处理结果：

```
I=imread('trees.tif');
I2=histeq(I);
figure,imshow(I2)
```

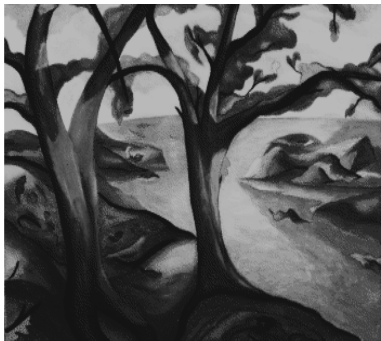


图 12-4 读取的原始图像





对图像 I2，再调用 imhist 函数可以观察均衡后的灰度值分布情况，如图 12-6 所示。语句如下：

```
figure,imhist(I2)
```



图 12-5 均衡后的图像

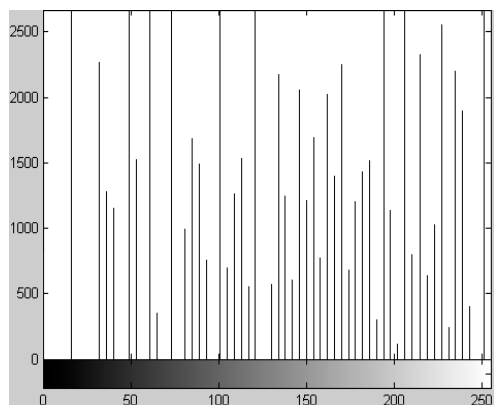


图 12-6 均衡后的图像灰度直方图

通过使用函数 histeq 来调节图像的像素分布，使之能够分布在与图像类型有关的整个取值范围内。对于一幅图像，如果存储类型是 unit8，那么相应的取值范围就是[0,256]；如果是 unit16，则取值范围是[0,65535]；如果是双精度类型，则取值范围是[0,1]。

比较图 12-3 和图 12-6 可知，I2 的直方图比 I 的直方图要长且平坦，这种铺展直方图的过程就叫直方图均衡化。

#### 4. 保存图像

下面将把均衡化后的图像 I2 保存到磁盘中。如果希望将该图像保存为 PNG 图像文件格式，则可以使用 imwrite 函数，并指定该保存图像的文件名和文件的扩展名 png，如以下命令行所示：

```
imwrite(I2,'trees2.png');
```

#### 5. 检查新生成文件的内容

保存图像后，如何知道在磁盘上写了什么内容呢？可以使用 imfinfo 函数来观察保存的图像文件信息。需要注意的是，在使用 imfinfo 函数时，不能在命令行末尾加上分号，这样才能确保 MATLAB 能够显示图像输出结果（只有 imfinfo 函数如此，其他显示函数加不加分号并没有影响，以下不再说明）。此外，还需保证此时的文件路径与调用 imwrite 时的路径一致。语句如下：

```
Imfinfo('trees2.png')
```

得到响应：

```
ans =  
Filename: 'trees2.png'  
FileModDate: '10-Mar-2008 00:00:26'
```

```

FileSize:55937
Format: 'png'
FormatVersion: []
Width: 350
Height:258
BitDepth: 8
ColorType: 'grayscale'
FormatSignature: [137 80 78 71 13 10 26 10]
    
```

### 12.1.3 图像处理的高级应用

通过前面的操作，读者对 MATLAB 的一些基本操作有了一定的了解，通过下面的练习，将掌握 MATLAB 的一些较为高级的操作。本小节练习主要目的是消除 rice.tif 图像中亮度不一致的背景，并使用阈值将修改后的图像转换为二值图像，使用成员标记返回图像中对象的个数以及统计特性。

(1) 读取和显示图像。清除 MATLAB 工作平台的所有变量，关闭已经打开的图形窗口，读取和显示灰度图像 rice.png，如图 12-7 所示。语句如下：

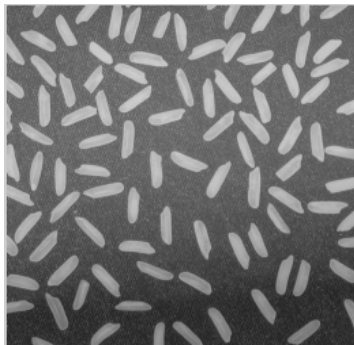


图 12-7 原始图像

```

clear,close all
I=imread('rice.png');
imshow(I)
    
```

(2) 估计图像背景。从图 12-7 可知，图像中心位置的背景亮度要高于其他部分的亮度。在 MATLAB 中，可以使用 imopen 函数和一个半径为 15 的圆盘形结构元素对输入的图像 I 进行形态学开操作，去掉那些不完全包括在圆盘中的对象，从而实现对背景亮度的估计。实现程序代码如下：

```

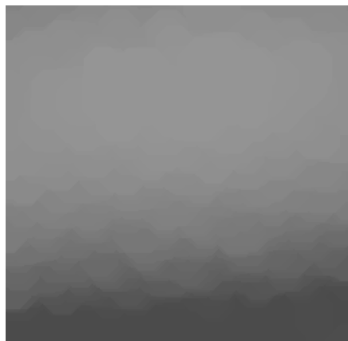
background=imopen(I,strel('disk',15));
imshow(background)
figure,surf(double(background(1:8:end,1:8:end))),zlim([0 256]);
set(gca, 'Ydir', 'reverse');
    
```

生成的背景图如图 12-8 (a) 所示，图像 12-8 (b) 所示则为以表面图形式显示的背景。

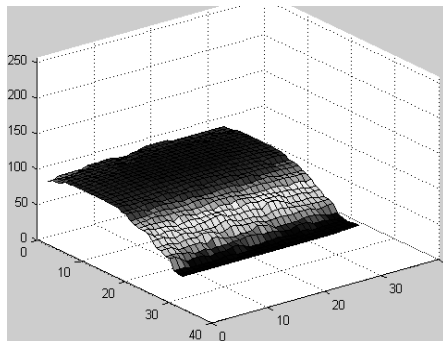
(3) 从原始图像中减去背景图像。将背景图像 background 从原始图像 I 中减去，从而创建一个新的、背景较为一致的图像，如图 12-9 所示。语句如下：

```

I2=imsubtract (I,background );
figure,imshow( I2 )
    
```



(a) 背景图



(b) 背景表面图

图 12-8 背景及背景表面图

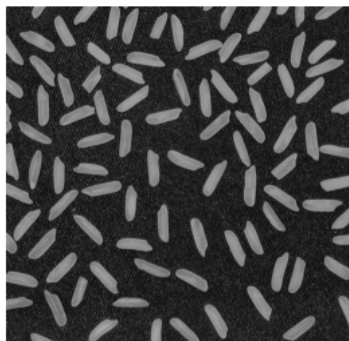


图 12-9 除去背景后的图像

(4) 调节图像对比度。从图 12-9 可以看出,修改后的图像比较暗,可以使用 MATLAB 提供的 `imadjust` 函数来调节图像的对比度。语句如下:

```
I3=imadjust(I2,stretchlim(I2),[0 1]);
figure,imshow(I3)
```

调节后的图像效果如图 12-10 所示。

(5) 使用阈值操作将图像转换为二进制图像。使用 `graythresh` 和 `im2bw` 函数可以创建一个新的二值图像 BW,如图 12-11 所示。语句如下:

```
level=graythresh(I3);
BW=im2bw(I3,level);
figure,imshow(BW)
```

调用 `whos` 命令可以查看图像的存储信息:

```
>> whos

Name      Size      Bytes    Class
BW        256x256   65536    logical array
I         256x256   65536    uint8 array
I2        256x256   65536    uint8 array
I3        256x256   65536    uint8 array
background 256x256   65536    uint8 array
level     1x1       8        double array
```

Grand total is 327681 elements using 327688 bytes

(6) 检查图像中的对象个数。为了确定图像中的米粒个数,可以使用 `bwlabel` 函数。该函数表示了二值图像 BW 中的所有相关成分,并且返回在图像中找到的对象个数。语句如下:

```
[labeled,numObjects]=bwlabel(BW,4)
numObjects =
    101
```

对于图 12-11 中有些相互连接的米粒，bwlabel 函数将它们视为同一个对象。

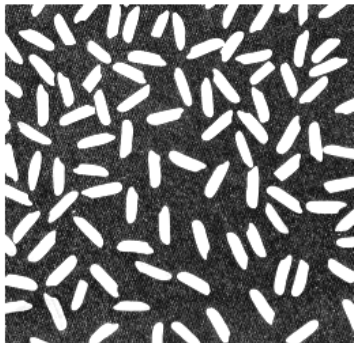


图 12-10 调节对比度后图像



图 12-11 二值化图像

（7）检查标记矩阵。可以使用 imcrop 命令来选择并显示已标记的对象和部分背景内的像素。可选择较小的矩形来进行这项操作，以保证显示的像素值不会引起 MATLAB 命令窗口的滚动。以下语句将使用 imcrop 函数进行交互式操作，当用户的鼠标位于图像范围内时，其形状会变成十字形，通过单击并拖动鼠标可选择标记区域，选择完成后，imcrop 函数将显示用户指定的标记区域（根据选择区域的不同，grain 显示矩阵是不同的）：

```
grain = imcrop(labeled)
grain =
    0     0     0    14
    0     0     0    14
    0     0     0     0
    0     0     0     0
```

观察标记矩阵的一个办法就是将其显示为一幅伪彩色的索引图像。在伪彩色的彩色图像中，标记矩阵中的每个对象都将被映射为相关调色板中的不同颜色，可以使用 label2rgb 色板中的对应颜色，处理效果如图 12-12 所示。语句如下：

```
RGB_label=label2rgb(labeled,@spring, 'c', 'shuffle');
imshow(RGB_label)
```

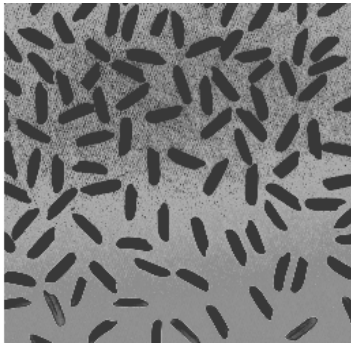


图 12-12 伪彩色索引图像

## 12.2 图像的小波分解和重构算法

### 12.2.1 二维小波变换及相应的快速算法

要处理二维函数或信号（如图像信号），就必须引入二维小波和二维小波变换及相应的快速算法。二维多分辨分析有两种，一种是可分离的，另一种是不可分离的。前一种情况简单且应用广泛。因此本小节就介绍可由一维多分辨分析的张量积空间构造的二维多分辨分析。



不可分离的情况也比较常见，但在图像处理领域应用不多，故这里不做介绍。

用  $L^2(R^2)$  表示平面上的平方可积函数空间，即

$$f(x, y) \in L^2(R^2) \Leftrightarrow \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} |f(x, y)|^2 dx dy < +\infty \quad (12-1)$$

容易证明，平面上有限区域中的一幅图像的能量是有限的。设  $f(x, y)$  是一幅图像，它的定义域围成的区域的面积为  $D$ ，设  $f(x, y)$  最大的亮度值为  $M$ ，即  $f(x, y) \leq M$ ，则

$$\int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} |f(x, y)|^2 dx dy \leq M^2 D < +\infty$$

引入  $L^2(R^2)$  空间的内积

$$\langle f, g \rangle = \int_{R^2} f(x, y) \overline{g(x, y)} dx dy, \quad f, g \in L^2(R^2)$$

相应的范数定义为

$$\|f\|_{L^2(R^2)} = \langle f, f \rangle^{1/2}, \quad f \in L^2(R^2)$$

在不发生混淆的情况下，范数也常记为  $\|f\|_{L^2}$ 。 $f(x, y)$  的傅里叶变换定义为

$$\hat{f}(\zeta) = \hat{f}(\zeta_1, \zeta_2) = \int_{R^2} f(x, y) e^{-i(x\zeta_1 + y\zeta_2)} dx dy$$

设  $F$  和  $D$  是两个有限维或可数无限维线性空间。 $F$  和  $D$  的基底分别为  $\dots, f_{-1}, f_0, f_1, \dots$  及  $\dots, d_{-1}, d_0, d_1, \dots$ 。定义以形如  $f_i d_j (i = 0, \pm 1, \pm 2, \dots; j = 0, \pm 1, \pm 2, \dots)$  的元素为基底的空间  $H$ ，为  $F$  与  $D$  的张量积空间，表示为

$$H = F \otimes D$$

如果  $F$  和  $D$  都是函数空间， $x$  和  $y$  分别是  $F$  和  $D$  中的自变量，则张量积空间  $H$  中的元素称为二维张量积函数或张量积曲面。

现在，设  $\{V_k^1\}$  和  $\{V_k^2\}$  是由尺度函数  $\phi^1(x)$  和  $\phi^2(y)$  生成的两个多分辨分析，则可以得到  $V_k^1$  和  $V_k^2$  的张量积空间

$$V_k = V_k^1 \otimes V_k^2$$

由于  $V_k^1$  的基底为  $\{2^{k/2} \phi^1(2^k x - j)\}$ ， $V_k^2$  的基底为  $\{2^{k/2} \phi^2(2^k y - l)\}$ ，所以  $V_k$  的基底为  $\{2^k \phi^1(2^k x - j) \phi^2(2^k y - l)\}$ 。

对于二元函数  $f(x, y)$ ，引入记号

$$f_{k;j,l}(x, y) = 2^k f(2^k x - j, 2^k y - l)$$

记  $\phi(x, y) = \phi^1(x) \phi^2(y)$ ，则  $\{\phi_{k;j,l}(x, y) : j, l \in Z\}$  是  $V_k$  的基底。这样， $\{V_k\}$  就形成  $L^2(R^2)$  中的一个多分辨分析， $\phi(x, y)$  就是相应的尺度函数。

设  $V_k^1$  关于  $V_{k+1}^1$  的补空间  $W_k^1$ ， $V_k^2$  关于  $V_{k+1}^2$  的补空间  $W_k^2$ ，即

$$V_{k+1}^1 = V_k^1 \dot{+} W_k^1, \quad V_{k+1}^2 = V_k^2 \dot{+} W_k^2$$

现在，设  $\psi^1(x)$  生成  $W_0^1$ ， $\psi^2(x)$  生成  $W_0^2$ ，即

$$W_0^1 := \text{clos}_{L^2(R)} \langle \psi^1(x - k) : k \in Z \rangle$$

$$W_0^2 := \text{clos}_{L^2(R)} \langle \psi^2(x - k) : k \in Z \rangle$$

这时有



$$\begin{aligned}
 V_{k+1} &= V_{k+1}^1 \otimes V_{k+1}^2 = (V_k^1 \dot{+} W_k^1) \otimes (V_k^2 \dot{+} W_k^2) \\
 &= V_k^1 \otimes V_k^2 \dot{+} V_k^1 \otimes W_k^2 \dot{+} W_k^1 \otimes V_k^2 \dot{+} W_k^1 \otimes W_k^2 \\
 &= V_k \dot{+} W_k
 \end{aligned} \quad (12-2)$$

其中,

$$\begin{aligned}
 W_k &= W_k^{(1)} + W_k^{(2)} + W_k^{(3)} \\
 W_k^{(1)} &= V_k^1 \otimes W_k^2, W_k^{(2)} = W_k^1 \otimes V_k^2, W_k^{(3)} = W_k^1 \otimes W_k^2
 \end{aligned}$$

同样, 由于  $V_k^1$  的基底为  $\{2^{k/2} \phi^1(2^k x - j)\}$ ,  $W_k^2$  的基底为  $\{2^{k/2} \psi^1(2^k y - l)\}$ , 则  $W_k^{(1)}$  的基底为  $\{2^k \phi^1(2^k x - j) \psi^2(2^k y - l)\}$ 。

记  $\psi^1(x, y) = \phi^1(x) \psi^2(y)$ , 则  $W_k^{(1)}$  的基底为  $\{\psi_{k;j,l}^1 : j, l \in Z\}$ 。

类似地, 记  $\psi^2(x, y) = \psi^1(x) \phi^2(y)$ ,  $\psi^3(x, y) = \psi^1(x) \psi^2(y)$ , 则  $W_k^{(2)}$  的基底为  $\{\psi_{k;j,l}^2 : j, l \in Z\}$ ,  $W_k^{(3)}$  的基底为  $\{\psi_{k;j,l}^3 : j, l \in Z\}$ 。

可以看到, 与一维只有 1 个尺度函数和 1 个小波函数不同的是, 二维情形有 1 个尺度函数  $\phi(x, y)$  和 3 个小波函数  $\psi^1(x, y)$ 、 $\psi^2(x, y)$ 、 $\psi^3(x, y)$ 。

与一维情况类似, 由式 (2-12), 我们有直和分解

$$L^2(R^2) = \dots \dot{+} W_{-1} \dot{+} W_0 \dot{+} W_1 \dot{+} \dots$$

则对于  $\forall f(x, y) \in L^2(R^2)$  都有唯一分解

$$f(x, y) = \dots + d_{-1}(x, y) + d_0(x, y) + d_1(x, y) + \dots$$

其中,  $d_k(x, y) \in W_k, k = -1, 0, 1, \dots$ 。

若  $\phi^1(x), \phi^2(y)$  及  $\psi^1(x), \psi^2(y)$  都是半正交尺度函数与半正交小波函数, 则上面的直和分解就可以变为正交和分解

$$L^2(R^2) = \dots \oplus W_{-1} \oplus W_0 \oplus W_1 \oplus \dots$$

此时,  $W_k \perp W_n, k = -1, 0, 1, \dots$  且  $k \neq n$ , 即

$$\langle d_k, d_n \rangle = 0, k \neq n$$

其中,  $d_k \in W_k, d_n \in W_n$ 。

设  $f_k(x, y) \in V_k, d_k(x, y) \in W_k$ , 则

$$f_{k+1}(x, y) = f_k(x, y) + d_k(x, y)$$

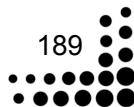
其中, 对于任何  $k, f_k \in V_k, d_k \in W_k$ 。这样, 对于  $d_k \in W_k$  还可以进一步分解为

$$d_k = d_k^{(1)} + d_k^{(2)} + d_k^{(3)}$$

其中,  $d_k^{(i)} \in W_k^{(i)} (i=1, 2, 3)$ 。则有  $f_{k+1}(x, y) \in V_{k+1}$ 。

利用一维情况下的两尺度方程和小波方程

$$\begin{cases}
 \phi^1(x) = \sum_n h_n^1 \phi^1(2x - n) \\
 \psi^1(x) = \sum_n g_n^1 \phi^1(2x - n) \\
 \phi^2(x) = \sum_n h_n^2 \phi^2(2x - n) \\
 \psi^2(x) = \sum_n g_n^2 \phi^2(2x - n)
 \end{cases}$$





可以得到二维张量积两尺度关系为

$$\begin{cases} \phi(x, y) = \sum_{n, m} h_{n, m} \phi(2x - n, 2y - m) \\ \psi^i(x, y) = \sum_{n, m} g_{n, m}^i \phi(2x - n, 2y - m), (i = 1, 2, 3) \end{cases}$$

其中,

$$\begin{cases} h_{n, m} = h_n^1 h_m^2, g_{n, m}^1 = h_n^1 g_m^2 \\ g_{n, m}^2 = g_n^1 h_m^2, g_{n, m}^3 = g_n^1 g_m^2 \end{cases}$$

设

$$\begin{aligned} f_k(x, y) &= \sum_{k; n, m} c_{k; n, m} \phi(2^k x - n, 2^k y - m) \\ g_k^{(i)}(x, y) &= \sum_{n, m} d_{k; n, m}^i \psi^i(2^k x - n, 2^k y - m) \end{aligned}$$

则由

$$f_{k+1}(x, y) = f_k(x, y) + g_k^{(1)}(x, y) + g_k^{(2)}(x, y) + g_k^{(3)}(x, y)$$

再利用尺度函数  $\phi(x, y)$  和小波函数  $\psi^1(x, y)$ 、 $\psi^2(x, y)$ 、 $\psi^3(x, y)$  及其二进伸缩和平移的正交性, 可以得到二维 Mallat 算法如下。

(1) 分解算法:

$$\begin{cases} c_{k+1; n, m} = \sum_{l, j} h_{l-2n} h_{j-2m} c_{k+1; l, j} \\ d_{k; n, m}^1 = \sum_{l, j} h_{l-2n} g_{j-2m} c_{k+1; l, j} \\ d_{k; n, m}^2 = \sum_{l, j} g_{l-2n} h_{j-2m} c_{k+1; l, j} \\ d_{k; n, m}^3 = \sum_{l, j} g_{l-2n} g_{j-2m} c_{k+1; l, j} \end{cases}$$

(2) 重构算法:

$$c_{k+1; n, m} = \sum_{l, j} h_{n-2l} h_{m-2j} c_{k; n, m} + \sum_{l, j} h_{n-2l} g_{m-2j} d_{k; n, m}^1 + \sum_{l, j} g_{n-2l} h_{m-2j} d_{k; n, m}^2 + \sum_{l, j} g_{n-2l} g_{m-2j} d_{k; n, m}^3$$

类似地, 利用一维双正交多分辨分析, 可以获得二维双正交多分辨分析。只要将相应的分解和重构滤波器置换, 就可以得到二维双正交多分辨分析的 Mallat 算法。我们称序列  $\{c^k, d_k^1, d_k^2, d_k^3\}$  为  $c^{k+1}$  的一级二维小波变换。

对应于二维 Mallat 算法的滤波器组表示如图 12-13 所示。

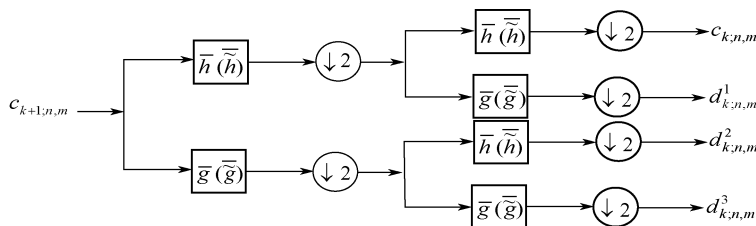
有了上面的分析, 现在就可以分析二维离散图像信号的处理方法。设  $\{b_{n, m}\} (n=0, 1, \dots, N-1)$  是一幅输入图像, 其像素点之间的距离为  $N^{-1}$ , 其中,  $N = 2^L$ 。我们可以将  $b_{n, m}$  与尺度  $2^L$  下的一个逼近函数

$$f(x, y) = \sum_{n, m} c_{n, m}^L \phi_{L, n, m}(x, y) \in V_L^2$$

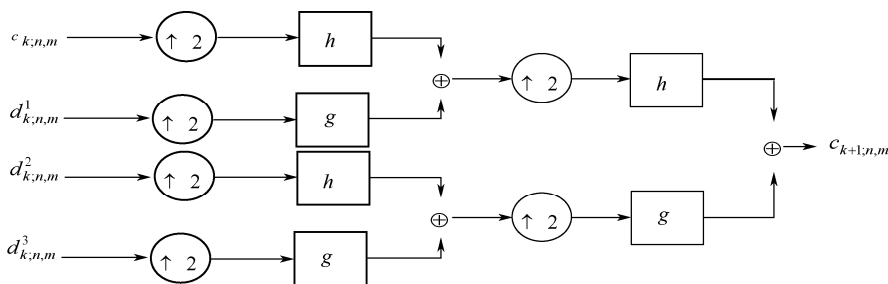
联系起来, 其中,  $c_{n, m}^L = \langle f, \tilde{\phi}_{L, n, m} \rangle$ ,  $\phi, \tilde{\phi}$  是两个对偶尺度函数。使得  $b_{n, m}$  为  $f(x, y)$  的均匀采样, 即  $b_{n, m} = f(N^{-1}n, N^{-1}m)$ 。另外, 根据  $c_{n, m}^L = \langle f, \tilde{\phi}_{L, n, m} \rangle$ , 有



$$Nc_{n,m}^L = \int_{-}^{+} \int_{-}^{+} f(u,v) \frac{1}{N^{-2}} \tilde{\phi}\left(\frac{u-N^{-1}n}{N^{-1}}, \frac{v-N^{-1}m}{N^{-1}}\right) du dv$$



(a) 二维小波分解 (括号中表示双正交滤波器)



(b) 二维小波重构

图 12-13 二维二通道 Mallat 算法的滤波器组表示

由于  $\int_{-}^{+} \int_{-}^{+} \phi(u,v) du dv = 1$ , 故

$$\int_{-}^{+} \int_{-}^{+} \frac{1}{N^{-2}} \tilde{\phi}\left(\frac{u-N^{-1}n}{N^{-1}}, \frac{v-N^{-1}m}{N^{-1}}\right) du dv = 1$$

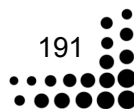
从而,  $Nc_{n,m}^L$  是  $f$  在  $(N^{-1}n, N^{-1}m)$  的一个小邻域上的加权平均。因此有

$$Nc_{n,m}^L \approx f(N^{-1}n, N^{-1}m) = b_{n,m}$$

若将  $\{c_{k+1;n,m}\}$  看成是一幅二维图像信号,  $n$  和  $m$  分别为行下标和列下标, 则二维小波变换过程可以解释为: 先利用分析滤波器  $\tilde{h}$ 、 $\tilde{g}$  对图像的每  $n$  行做小波变换, 得到低频部分  $\sum_j \tilde{h}_{j-2m} c_{k+1;l,j}$  和高频部分  $\sum_j \tilde{g}_{j-2m} c_{k+1;l,j}$ ; 然后对得到的数据的每  $m$  列用分析滤波器  $\tilde{h}$ 、 $\tilde{g}$  做小波变换; 对  $\sum_j \tilde{h}_{j-2m} c_{k+1;l,j}$  的各列做小波变换得到低频系数  $\sum_l \tilde{h}_{l-2n} (\sum_j \tilde{h}_{j-2m} c_{k+1;n,m})$ , 即  $c_{j;n,m}$ ; 得到高频系数  $\sum_l \tilde{g}_{l-2n} (\sum_j \tilde{h}_{j-2m} c_{k+1;n,m})$ , 即  $d_{k;n,m}^1$ ; 对  $\sum_j \tilde{g}_{j-2m} c_{k+1;l,j}$  的各列做小波变换得到低频系数  $\sum_l \tilde{h}_{l-2n} (\sum_j \tilde{g}_{j-2m} c_{k+1;n,m})$ , 即  $d_{k;n,m}^2$  及高频系数  $\sum_l \tilde{g}_{l-2n} (\sum_j \tilde{g}_{j-2m} c_{k+1;n,m})$ , 即  $d_{k;n,m}^3$ 。一级小波分解后图像由四部分构成, 即

$$\begin{pmatrix} (c_{k;n,m}) & (d_{k;n,m}^1) \\ (d_{k;n,m}^2) & (d_{k;n,m}^3) \end{pmatrix}$$

其中, 每个子图像都是原始图像尺寸大小的 1/4。这样, 每级变换得到的低频信号递归地进行







分解。重构过程也可类似地进行。这样就形成了二维小波变换的塔式结构。

回顾从一维离散小波变换到二维的扩展，二维静态小波变换采用相似的方式，对行和列分别采用高通和低通滤波器。这样分解的结果仍然是四组图像、近似图像、水平细节图像、竖直细节图像和对角图像，与离散小波变换不同的只是静态小波分解得到的四幅图像与原图像尺寸一致，道理与一维情况相同。

## 12.2.2 小波分解和重构 MATLAB 例程

二维离散小波变换同样只提供了一个函数 `swt2`，因为它不对分解系数进行下采样，所以单层分解和多层分解的结果是一样的，不需要另外提供多层分解的功能。

下面举一个使用 `swt` 命令的例子，大家可以对比它和 `dwt` 处理结果的区别。

**【例 12-1】** 小波分析用于图像分解。

在命令行下输入：

```
load noiswom
[swa,swh,swv,swd]=swt2(X,3,'db1');
% 使用 db1 小波对 noiswom 图像进行 3 层静态小波分解
whos
% 可以看出，swt2 所小波分解同样不改变信号的长度，原来的 96×96 的图
% 像做了 3 层分解以后，分解系数是 12 个 96×96 的图像。
colormap(map)
kp=0;
for i=1:3
    subplot(3,4,kp+1),image(wcodemat(swa(:,i),192));
    title(['Approx.cfs level',num2str(i)])
    % 显示第 i 层近似系数图像，以 192 字节为单位编码
    subplot(3,4,kp+2),image(wcodemat(swh(:,i),192));
    title(['Horiz.Det.cfs level',num2str(i)])
    subplot(3,4,kp+3),image(wcodemat(swv(:,i),192));
    title(['Vert.Det.cfs level',num2str(i)])
    subplot(3,4,kp+4),image(wcodemat(swd(:,i),192));
    title(['Diag.Det.cfs level',num2str(i)])
    kp=kp+4;
end
```

显示的结果如图 12-14 所示，由于分解过程中没有改变信号的长度，所以在显示近似和细节系数时不需要重建。

同 `idwt2` 类似，MATLAB 对二维静态小波重建提供了 `iswt2` 命令，`idwt` 的去边也同一维情况类似，对经过重建滤波后的信号不进行上采样（因为近似系数和细节系数大小都与原信号一致）。

同一维的静态小波重建一样，我们将举例子说明如何将 `iswt2` 单纯用作滤波器来实现各层系数的重建，与一维的情况不同的只是为了重建第  $j$  层近似系数，需要 4 次用到 `iswt2` 作为重



建滤波器对第  $j+1$  层的系数进行滤波，在对某个近似系数滤波的过程中，同样需要把其他 3 个系数指定为 0。

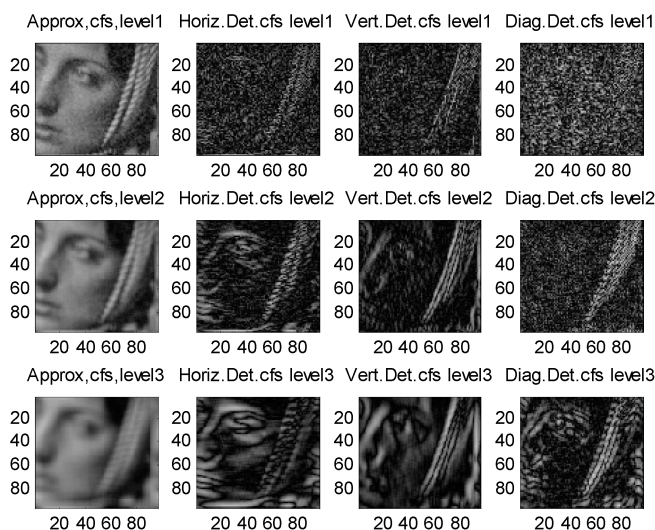
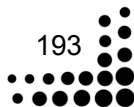


图 12-14 小波分析用于图像分解

为了便于比较，本例直接利用例 12-1 中对 noiswom 的分解结果，从中重建各级系数。

【例 12-2】db1 小波下各级静态小波重建系数。

```
load noiswom
[swa,swh,swv,swd]=swt2(X,3,'db1');
% 使用 db1 小波对 noiswom 图像进行 3 层小波分解
mzero=zeros(size(swd));
A=mzero;
A(:,:,3)=iswt2(swa,mzero,mzero,mzero,'db1');
% 使用 iswt2 的滤波器功能，重建第 3 层的近似系数，为了避免 iswt 的合
% 成运算，注意在重建过程中，应保证其他各项系数为零
H=mzero;V=mzero;D=mzero;
for i=1;3
    swcfs=mzero;swcfs(:,i)=swh(:,i);
    H(:,i)=iswt2(mzero,swcfs,mzero,mzero,'db1');
    swcfs=mzero;swcfs(:,i)=swv(:,i);
    V(:,i)=iswt2(mzero,mzero,swcfs,mzero,'db1');
    swcfs=mzero;swcfs(:,i)=swd(:,i);
    H(:,i)=iswt2(mzero,mzero,mzero,swcfs,'db1');
end
% 分别重建 1~3 级的各个细节系数，同样在重建某一系数的时候，要令其他系数为 0
A(:,:,2)=A(:,:,3)+H(:,:,3)+V(:,:,3)+D(:,:,3);
A(:,:,1)=A(:,:,2)+H(:,:,2)+V(:,:,2)+D(:,:,2);
% 使用递推的方法建立第 1 层和第 2 层的近似系数
```





```

colormap(map)
kp=0;
for i=1:3
subplot(3,4,kp+1),image(wcodemat(A(:,i),192));
title(['第',num2str(i),'层近似系数图像'],'fontsize',6)
subplot(3,4,kp+2),image(wcodemat(H(:,i),192));
title(['第',num2str(i),'层水平细节系数图像'],'fontsize',6)
subplot(3,4,kp+3),image(wcodemat(V(:,i),192));
title(['第',num2str(i),'层竖直细节系数图像'],'fontsize',6)
subplot(3,4,kp+4),image(wcodemat(D(:,i),192));
title(['第',num2str(i),'层对角细节系数图像'],'fontsize',6)
kp=kp+4;
end
% 画出通过手工方法重建的各级小波系数图像
err=norm(A(:,2)-swa(:,2))
% 求出用这种算法重建的第2层近似系数和分解系数之间的误差
err =
2.9297e+004

```

显示的结果如图 12-15 所示。

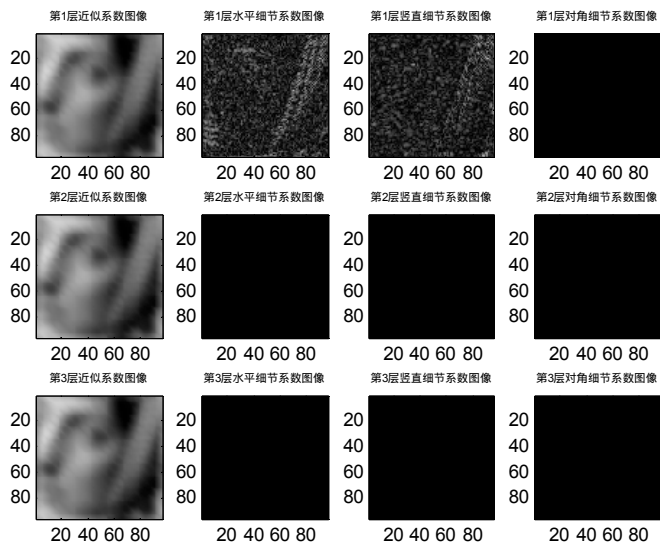


图 12-15 在 db1 小波下各级静态小波重建系数

## 第 13 章 函数的奇异性与 故障信号检测分析

信号的奇异点能够通过对信号进行小波变换后在不同尺度上的综合表现，来反映信号的突变或者瞬态特征。本章在分析电力系统暂态故障信号的奇异性和小波变换奇异性检测基本原理的基础上，得出电力系统暂态故障信号奇异性的特殊性，提出了利用小波变换进行故障暂态信号奇异性检测的方法。由于不同的小波类型对突变信号的检测有不同的效果，故提出了一种选择小波类型的方法。

### 13.1 故障信号检测的理论分析

如果一个函数  $f$  在  $t_0$  点不可微，则说它在  $t_0$  点是奇异的。现将 Lipschitz 指数引申到  $0 < \alpha < 1$ ，用以度量函数的奇异性。

#### 13.1.1 函数的奇异性

定义 令  $0 < \alpha < 1$ ，如存在一个常数  $C$ ，使

$$\forall t \in R, |f(t) - f(t_0)| \leq C |t - t_0|^\alpha \quad (13-1)$$

成立，则称  $f$  在点  $t_0$  是 Lipschitz  $\alpha$  的。如对所有的  $t_0 \in [a, b]$  和一个与  $t_0$  无关的常数  $C$ ，使得式 (13-1) 成立，则称  $f$  在区间  $[a, b]$  是一致 Lipschitz  $\alpha$  的。 $\alpha$  的上界值称为 Lipschitz 奇异性。

若  $f$  在点  $t_0$  可微，则其 Lipschitz 指数至少为 1，粗略地说，如  $\alpha = 1$ ，则式 (13-1) 可改写为  $\left| \frac{f(t) - f(t_0)}{t - t_0} \right| \leq C$ ，当  $t$  趋近于  $t_0$  时，该不等式的左边实际上就是  $f$  在  $t_0$  点的一阶导数  $f'(t_0)$ ，取  $C = |f'(t_0)|$ ，则式 (13-1) 成立。

若  $f$  在点  $t_0$  不连续但在  $t_0$  的邻域有界，或者说它在  $t_0$  有有限跃度，则其 Lipschitz 指数为 0。当  $\alpha = 0$  时，式 (13-1) 成为  $|f(t) - f(t_0)| \leq C$ ，该不等式左边最大值等于  $f$  在  $t_0$  点的跃度，取  $C$  等于或大于跃度，则式 (13-1) 成立。

还可以将 Lipschitz 指数推广到为负数的情况。若  $f$  的原函数在  $t_0$  点的 Lipschitz 指数为  $\alpha$ ，



则  $f$  在该点的 Lipschitz 指数为  $\alpha - 1$ 。例如,  $\delta(t - t_0)$  的原函数为一单位阶跃, 它在  $t_0$  的 Lipschitz 指数为 0, 故  $\delta(t - t_0)$  在  $t_0$  点的 Lipschitz 指数为 -1。现在可以清楚地看到, Lipschitz 指数确实能在更一般的意义下定量地描述函数的奇异性。

为了能用小波变换准确检测故障信号的位置, 希望小波有小的支集, 即希望小波在时域快速衰减, 这意味着对任意衰减指数  $m \in N$ , 存在  $C_m$ , 使得

$$\forall t \in R, |\psi(t)| \leq \frac{C_m}{1 + |t|^m} \quad (13-2)$$

若小波有  $K$  阶消失距, 则它与直到  $K - 1$  次的多项式正交。若函数  $f$  在  $t_0$  点足够正则, 那么小波变换将取决于函数的  $K$  阶导数。下面的定理明确指出, 小波变换确实相当于一个微分算子。

### 13.1.2 Lipschitz 指数分析

定理 1 快速衰减的小波  $\psi$  具有  $K$  阶消失距, 当且仅当在快速衰减的函数  $\theta$ , 使得

$$\psi(t) = (-1)^K \frac{d^K}{dt^K} \theta(t) \quad (13-3)$$

从而

$$Wf(s, t) = s^K \frac{d^K}{dt^K} (f * \bar{\theta}_s)(t) \quad (13-4)$$

其中,  $\bar{\theta}_s(t) = s^{-1/2} \theta(-t/s)$ 。而且  $\psi$  具有不超过  $K$  阶的消失距, 当且仅当  $\int_{-\infty}^{+\infty} \theta(t) dt \neq 0$ 。

式 (13-4) 意味着小波变换等于用  $\bar{\theta}_s(t)$  平滑信号后求  $K$  阶导数, 所以它相当于一个多尺度微分算子, 式中的小波变换为通常使用的相关型小波变换, 而且本节都假定小波是实小波, 即

$$Wf(s, t) = \frac{1}{\sqrt{s}} \int_{-\infty}^{+\infty} f(\tau) \psi\left(\frac{\tau - t}{s}\right) d\tau = f * \bar{\psi}_s(t) \quad (13-5)$$

式中,  $s \in R^+$  为尺度参数;  $t \in R$  为平移参数;  $\bar{\psi}_s(\tau) = s^{-1/2} \psi(-\tau/s)$ 。

由于  $\theta$  的积分不等于零, 所以它是一个平滑函数。例如, 高斯函数就是一个经常用到的平滑函数, 其数学表达式为

$$\theta(t) = \frac{1}{\sqrt{\pi}} e^{-t^2}$$

其中, 常数  $1/\sqrt{\pi}$  使  $\int_{-\infty}^{+\infty} \theta(t) dt = 1$ 。高斯函数的 1 阶导数和 2 阶导数分别为

$$\psi^1(t) = -2\left(\frac{2}{\pi}\right)^{\frac{1}{4}} t e^{-t^2} \text{ 和 } \psi^2(t) = -\frac{2}{\sqrt{3}}\left(\frac{2}{\pi}\right)^{\frac{1}{4}} (1 - 2t^2) e^{-t^2}$$

上两式中的常数项是使其范数等于 1。

定理 2 设  $f(t) \in L^2(R)$  在区间  $[a, b]$  是一致 Lipschitz  $\alpha$   $K$  的, 总存在  $A > 0$ , 使得



$$\forall (s, t) \in R^+ \times [a, b], |Wf(s, t)| \leq As^{\alpha+1/2} \quad (13-6)$$

反之, 若  $f$  有界且对某一非整数的  $\alpha < K$ ,  $Wf(s, t)$  满足上式, 则对  $\forall \varepsilon > 0$ ,  $f$  在  $[a + \varepsilon, b - \varepsilon]$  上是一致 Lipschitz  $\alpha$  的。

式 (13-6) 中的小波变换是相关型小波变换。在 Mallat 早期关于奇异性检测的文献中, 用的是卷积型小波变换, 这时式 (13-6) 的右边修改为  $As^\alpha$  即可。

上述定理实际上是  $s \rightarrow 0$  时,  $|Wf(s, t)|$  衰减速度与 Lipschitz 指数的关系, 对大尺度、Cauchy-Schwarz 不等式指出小波变换是有界的, 即

$$|Wf(s, t)| = |\langle f, \psi_{s,t} \rangle| \leq \|f\| \cdot \|\psi\|$$

这时小波变换的模将不受式 (13-6) 的约束。

上述定理是关于区间 Lipschitz 指数的, 下述定理则将一点的 Lipschitz 指数与小波变换的衰减速度联系起来。

定理 3 设  $f(t) \in L^2(R)$  在  $t_0$  点是 Lipschitz  $\alpha < K$  的, 总存在  $A > 0$ , 使得

$$\forall (s, t) \in R^+ \times R, |Wf(s, t)| \leq As^{\alpha+1/2} \left( 1 + \left| \frac{t-t_0}{s} \right|^\alpha \right) \quad (13-7)$$

反之, 设非整数的  $\alpha < K$ , 且存在  $A$  和  $\alpha' < \alpha$ , 使得

$$\forall (s, t) \in R^+ \times R, |Wf(s, t)| \leq As^{\alpha+1/2} \left( 1 + \left| \frac{t-t_0}{s} \right|^{\alpha'} \right) \quad (13-8)$$

则  $f$  在  $t_0$  点是 Lipschitz  $\alpha$  的。

若  $\psi$  是紧支的, 支集为  $[-c, c]$ , 则  $s^{-1/2}\psi(t/s)$  的支集为  $[-sc, sc]$ 。在时间-尺度平面上, 当时间平移量  $t$  满足

$$|t - t_0| \leq cs \quad (13-9)$$

时, 分析小波的支集才包含  $t_0$  点。式 (13-9) 规定了时间-尺度平面上的一个锥形区域, 称为  $t_0$  点的影响锥。在影响锥内, 有  $|t - t_0|/s \leq c$ , 这时式 (13-7) 和式 (13-8) 可写成

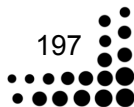
$$|Wf(s, t)| \leq A's^{\alpha+1/2}$$

因为幅值较大的小波系数落在奇异点的影响锥内, 所以用小波变换度量 Lipschitz 指数时, 定理 3 与定理 2 的一致 Lipschitz 条件完全相同。

上面两个定理指出, 随着尺度越来越精细, 小波变换模将呈指数衰减, Lipschitz 指数越大, 衰减越快。这样, 就可以由小变换的衰减速度来度量 Lipschitz 指数。

定理 4 设  $\psi$  是紧支的  $C^K$  函数, 且  $\psi = (-1)^K \theta^{(K)}$ ,  $\int_{-\infty}^{+\infty} \theta(t) dt \neq 0$ , 设  $f \in L^1[a, b]$ , 若存在  $s_0 > 0$ , 以致  $t \in [a, b]$  及  $s < s_0$  时,  $|Wf(s, t)|$  没有局部极大, 则对任意  $\varepsilon > 0$ , 函数  $f$  在  $[a - \varepsilon, b + \varepsilon]$  上是一致 Lipschitz  $K$  的。

定理 4 并不保证出现模极大值线的点就一定是奇异点, 实际情况也确实是这样。所以在出现模极大值线的情况下, 应进一步计算 Lipschitz 指数, 从而判断该点是否奇异, 如该点确实是奇异点, 则可用 Lipschitz 指数衡量该点奇异的性质。





## 13.2 实验结果与分析

### 13.2.1 利用小波分析检测传感器故障

电力系统在线监测信号含有大量的现场背景噪声，给传统方式的数据采集与故障诊断带来很大的困难，将以处理瞬态信号、含宽带噪声信号等见长的小波分析应用于电力系统在线监测是大有前途的。

采集的电力负载信号波形如图 13-1 (a) 所示，从图中可以看出，两个黑圆圈之间的信号出现异常，这是由传感器故障造成的。为了更清楚地揭示这种故障，利用 Daubechies 3 小波对其进行 3 层分解，得到 1~3 层的细节信号。可以看出，细节信号[见图 13-1 (b)~图 13-1 (d)] 都显示时间 2400~3600 之间的信号由于传感器故障而引入了传感器误差噪声。第 1 层分解的 d1 高频系数重构的图像比 d2 和 d3 高频系数重构的图像更清楚地确定了故障信号所在位置。

【例 13-1】利用小波分析检测传感器故障。

```
%装载采集的信号 leleccum.mat
load leleccum;
%将信号中第 2200~3600 个采样点值赋给 s
index=2200:3600;
s=leleccum(index);
%画出原始信号
figure;
plot(index,s);
title('(a)电力负载信号波形');
%用 db3 小波进行 5 层分解
[c,l]=wavedec(s,5,'db3');
%重构第 1~5 层细节系数
d5 = wrcoef('d',c,l,'db3',5);
d4 = wrcoef('d',c,l,'db3',4);
d3 = wrcoef('d',c,l,'db3',3);
d2 = wrcoef('d',c,l,'db3',2);
d1 = wrcoef('d',c,l,'db3',1);
%显示细节系数
figure;
plot(index,d3,'LineWidth',2);
title('(b)细节信号波形 d3');
figure;
plot(index,d2,'LineWidth',2);
title('(c)细节信号波形 d2');
figure;
```

```
plot(index,d1,'LineWidth',2);
title('(d)细节信号波形 d1');
```

程序运行结果如图 13-1 所示。

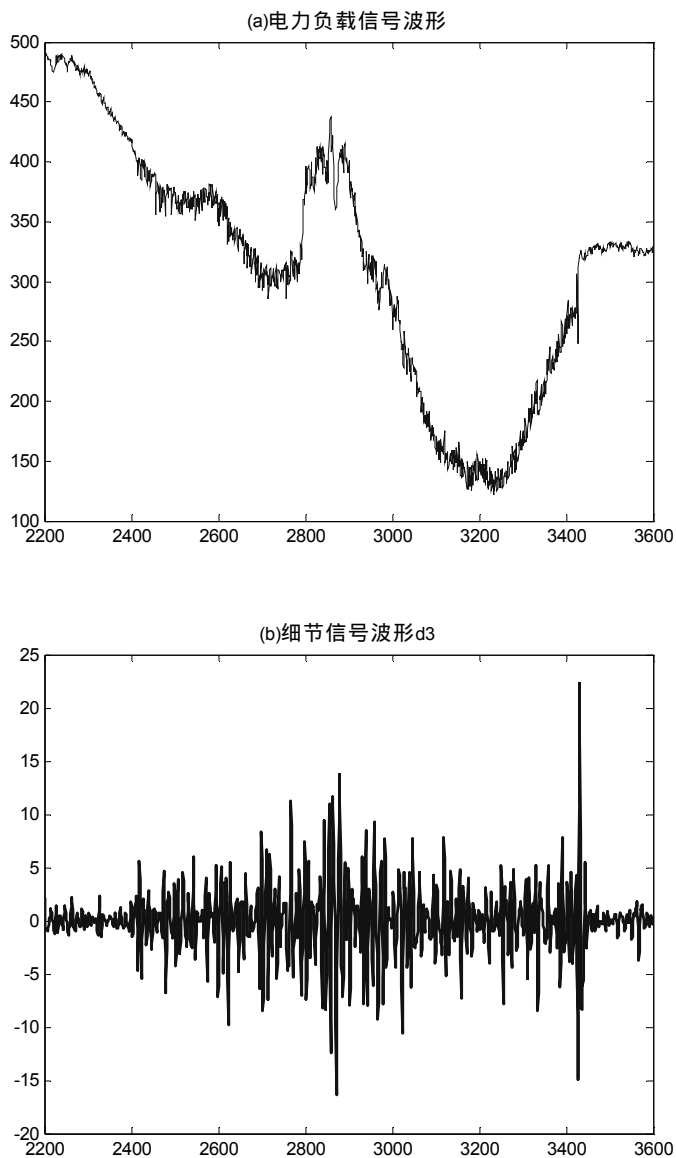


图 13-1 小波变换检测传感器故障示意图



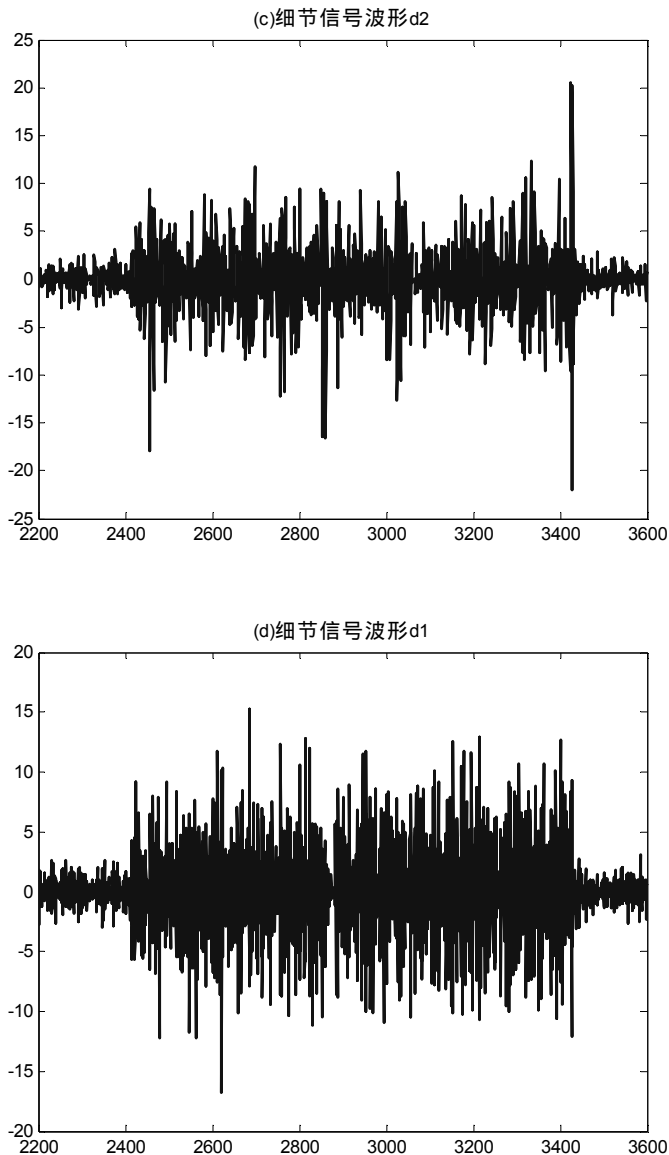


图 13-1 小波变换检测传感器故障示意图 (续)

【例 13-2】某正在工作的系统，正常工作输出点的采样信号应为蠕变信号，当系统出现故障时，输出信号会出现一个突变信号（主要表现在幅度和频率的突变）。现给出一个该系统从正常到出现故障的采样序列，请利用小波分析分析故障出现的时间点。

该问题是一个检测突变点（或不连续点）的问题，利用小波分析可以精确地检测出信号突变的时间点。

利用小波分析检测信号突变点的一般方法是，对信号进行多尺度分析，在信号出现突变时，其小波变换后的系数具有模极大值，因而可以通过对模极大值点的检测来确定故障发生的时间点。



程序清单如下：

```

t=0:pi/125:4*pi;
s1=sin(t);                                %设置一正常信号
s2=sin(10*t)                              %设置一故障信号，表现在频率的突变
s3=sin(t)                                %设置一正常信号
s=[s1,s2,s3];                             %整个信号

subplot(421);plot(s)
title('原始信号');
ylabel('s');
[c,l]=wavedec(s,6,'db3');                % 采用 db3 小波并对信号进行 6 层分解
apcmp=wrcoef('a',c,l,'db3',6);
subplot(422);plot(apcmp);
ylabel('ca6');
for i=1:6
    decmp=wrcoef('d',c,l,'db3',7-i);
    subplot(4,2,i+2);
    plot(decmp);
    ylabel(['d',num2str(7-i)]);
end
end

```

输出结果如图 13-2 所示。

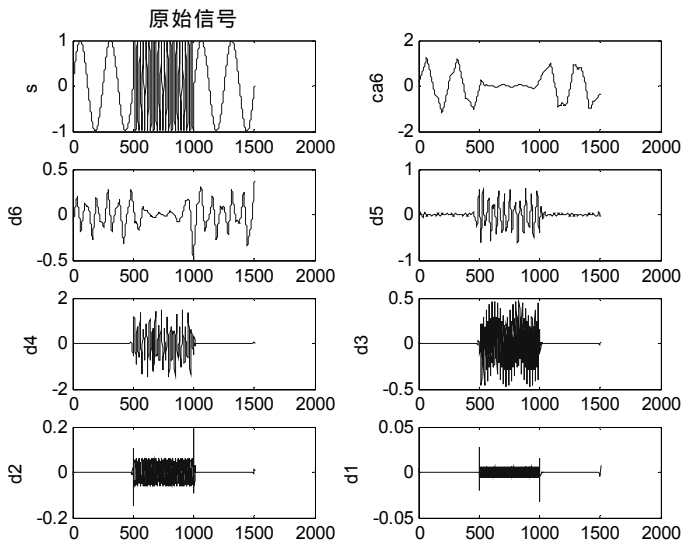
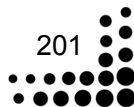


图 13-2 检测第一种类型的间断点

从图 13-2 中小波分解的层系数可以明显看出，在时间为 500 时，系统工作出现了异常情况；在时间为 1000 时，系统工作又恢复了正常。

这是一个利用小波分析检测信号突变点的典型实例。从该例可以看出，小波分析在检测





信号突变点（奇异点）上具有比傅里叶变换无法比拟的优越性，利用小波分析可以精确地检测出信号突变的时间点。

### 13.2.2 小波类型的选择对于检测突变信号的影响

不同的小波类型对突变信号的检测有不同的效果，对图 13-3 (a) 所示的突变信号分别采用 Haar 和 Daubechies 5 小波对信号进行处理，可以有到其效果的不同。其中，图 13-3 (b) ~ 图 13-3 (d) 所示不是用 Haar 小波处理后得到的高频信号；图 13-3 (f) ~ 图 13-3 (h) 所示是用 Daubechies 5 小波处理后的高频信号；图 13-3 (e) 所示是用 Daubechies 5 小波处理后的近似信号。

【例 13-3】小波类型的选择对于检测突变信号的影响。

```
clear;
load nearbrk;
whos;
figure;
plot(nearbrk);
title('(a) 频率突变信号');
[c,l]=wavedec(nearbrk,3,'haar');
a3=wrcoef('a',c,l,'haar',3);
d3=wrcoef('d',c,l,'haar',3);
d2=wrcoef('d',c,l,'haar',2);
d1=wrcoef('d',c,l,'haar',1);
[c,l]=wavedec(nearbrk,3,'db5');
aa3=wrcoef('a',c,l,'db5',3);
dd3=wrcoef('d',c,l,'db5',3);
dd2=wrcoef('d',c,l,'db5',2);
dd1=wrcoef('d',c,l,'db5',1);
%figure;
%plot(a3);
%title('(e) 近似信号 a3');
figure;
plot(d3);
title('(b) 细节信号 d3');
figure;
plot(d2);
title('(c) 细节信号 d2');
figure;
plot(d1);
title('(d) 细节信号 d1');
figure;
plot(aa3);
title('(e) 近似信号 a3');
```



```
figure;  
plot(dd3);  
title('(f) 细节信号 d3');  
figure;  
plot(dd2);  
title('(g) 细节信号 d2');  
figure;  
plot(dd1);  
title('(h) 细节信号 d1');
```

程序运行结果如图 13-3 所示。

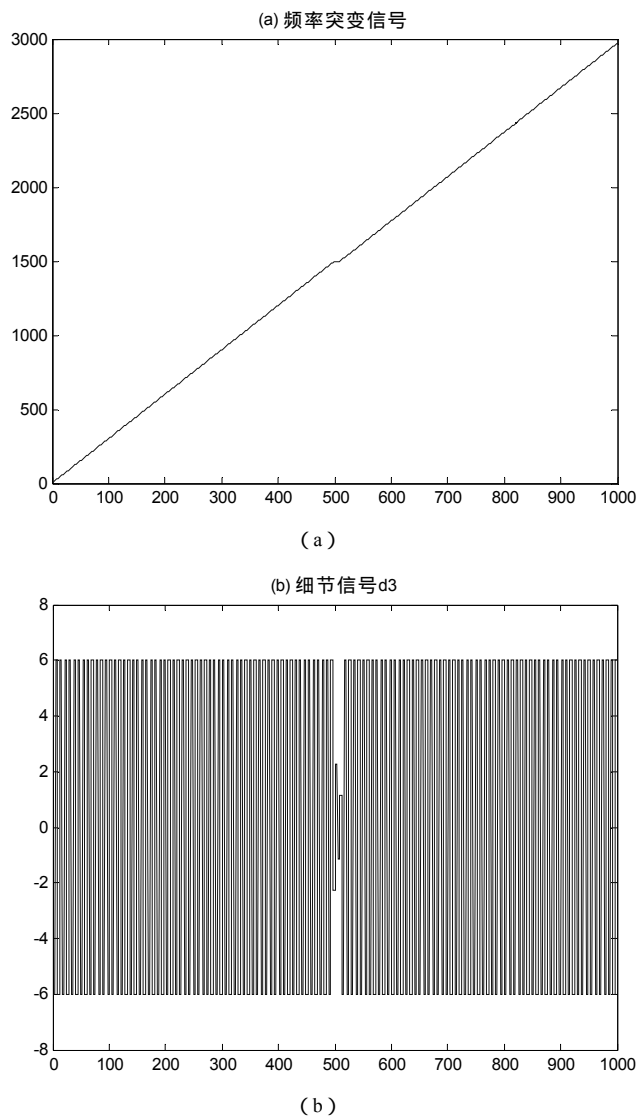
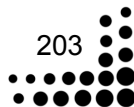


图 13-3 不同的小波类型检测突变信号的对比图



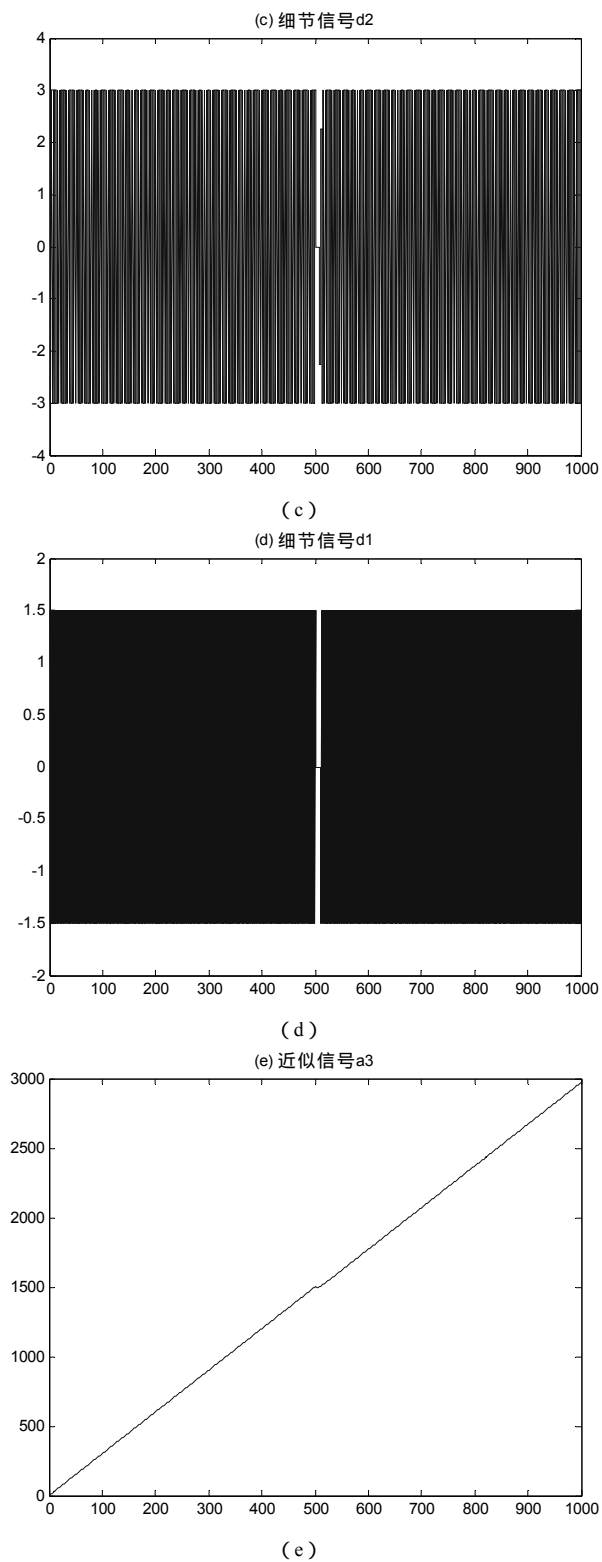


图 13-3 不同的小波类型检测突变信号的对比图 (续一)

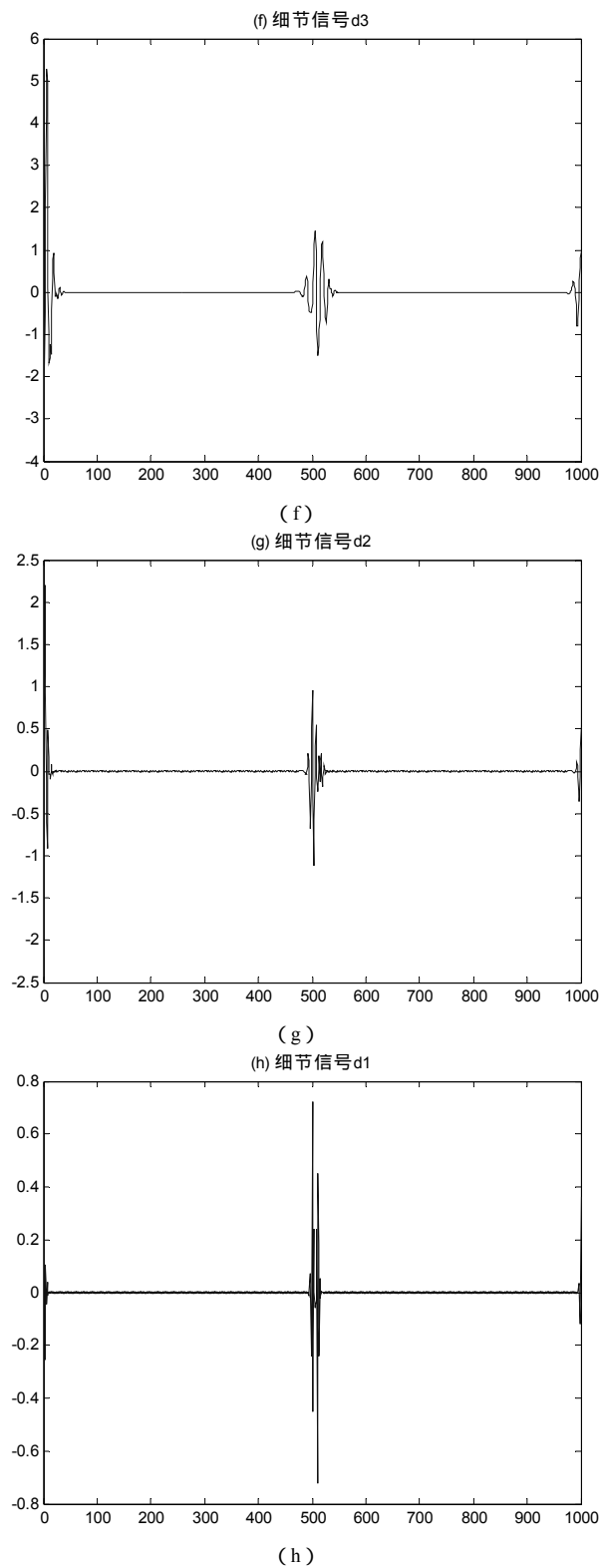


图 13-3 不同的小波类型检测突变信号的对比图 (续二)



由图 13-3 可见, Daubechies 5 小波分解后的 3 层高频系数重构图形可清楚地确定突变点的位置在时间为 500 处, 而 Haar 小波却没有这种能力。观察图 13-3 (b) ~ 图 13-3 (d) 可以发现, 由于在时间为 500 的点的系数值为零, 所以不能判断突变点的位置, 黑色的阴影说明小波系数在这段区间剧烈地振荡。这些振荡的结果是原信号与近似系数的差值, 因为 Haar 小波是分段的常数, 所以与连续信号相减时就会产生振荡。从图 13-3 同样可以看出, Daubechies 5 第 1 层分解的 d1 高频系数重构的图像比 d2、d3 高频系数重构的图像更清楚地确定了信号的突变点的位置。

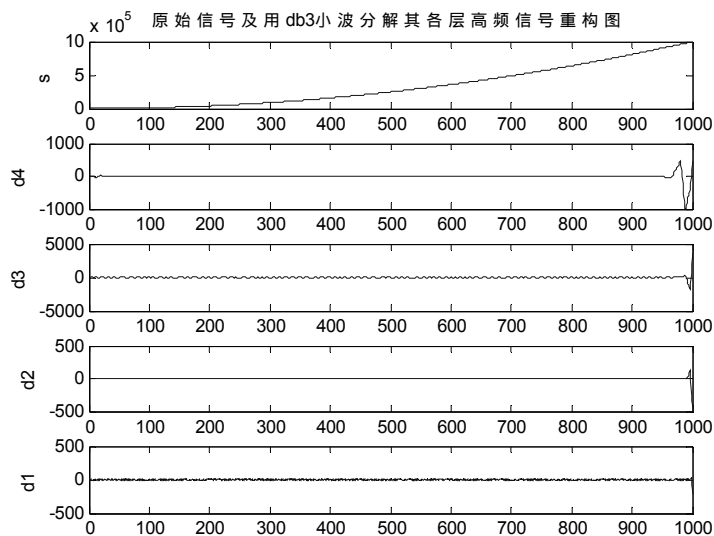
【例 13-4】给定一信号, 请利用 db3 和 db1 小波进行分析。

程序清单如下:

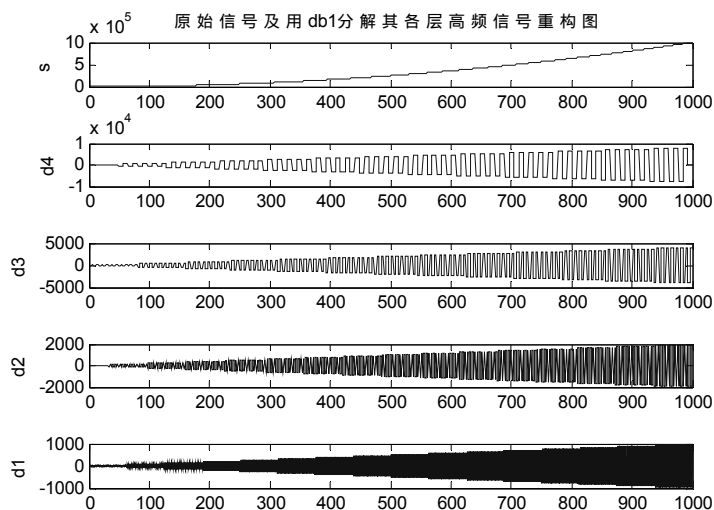
```
load noispol;                % 装入要分析的信号
s=noispol; ls=length(s);
[c,l]=wavedec(s,4,'db3');    % 用 db3 小波分解信号到第 4 层
figure(1);
subplot(5,1,1); plot(s);
title('原始信号及用 db3 小波分解其各层高频信号重构图');
ylabel('s');
% 对分解结构[c,l]中各高频部分进行重构, 并显示结果
for i=1:4
    decmp=wrcoef('d',c,l,'db3',5-i);
    subplot(5,1,i+1);
    plot(decmp)
    ylabel(['d',num2str(5-i)]);
end

% 下面用 db1 小波进行对照分析
[c,l]=wavedec(s,4,'db1');    % 用 db1 小波分解信号到第 4 层
figure(2);
subplot(5,1,1); plot(s);
title('原始信号及用 db1 小波分解其各层高频信号重构图');
ylabel('s');
% 对分解结构[c,l]中各高频部分进行重构, 并显示结果
for i=1:4
    decmp=wrcoef('d',c,l,'db1',5-i);
    subplot(5,1,i+1);
    plot(decmp);
    ylabel(['d',num2str(5-i)]);
end
```

输出结果如图 13-4 所示。



(a)



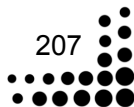
(b)

图 13-4 利用 db3 和 db1 小波进行分析比较

从 db3 小波和 db1 小波的对照可以看出, 由于 db3 小波有 3 个过零点, 而 db1 小波没有过零点, 其分解的高频系数表现出明显不同的特性。

### 13.3 小波类型选择

(1) 小波变换用于信号的突变点的检测, 无论采用小波变换系数的模极大点还是过零点方法, 都只有在多尺度上做综合分析和判断, 才能够准确地确定突变点的位置。通常, 较小







尺度下的小波变换能够减小频率混叠现象，判断突变点位置的准确度较高。

(2) 如何选择小波函数目前还没有一个理论标准，但是小波变换的小波系数为如何选择小波函数提供了依据。小波变换后的小波系数表明了小波与被处理信号之间的相似程度，如果小波变换后的小波系数比较大，就表明小波和信号的波形相似程度较大反之则比较小。另外，还要根据信号处理的目的是来决定尺度的大小。如果小波变换仅仅要反映信号整体的近似特性，则往往选用较大的尺度；如果反映信号细节上的变换，则选用尺度较小的小波。

## 第14章 利用提升小波算法 实现多分辨分析

随着多媒体应用领域的快速发展，新一代静止图像压缩标准——JPEG 2000 已在 2000 年 11 月完成了制定。该标准采用基于提升的离散小波变换作为其时频分析的核心，使得提升小波变换成为研究的热点。提升算法具有许多良好的特性：不依赖于傅里叶变换，有效避免了传统的基于卷积的算法中的冗余计算，降低了运算复杂度；可进行基于 in-place 存储的运算，减少了存储器开销；正反变换的硬件架构是完全相同的，反变换只是正变换反向的一个操作；内在并行处理结构提高了运算速度等。提升小波这种易于硬件实现的优点，使得对于基于提升格式小波变换的硬件电路结构设计得到了广泛研究。

经典小波分析是从傅里叶分析的基础上发展出来的，因而它在一定程度上受到傅里叶分析的限制。小波分析中的两个核心的概念——小波分析和多分辨分析都是建立在二进平移和伸缩思想基础之上的，这种思想直接来源于信号处理领域。我们称这种经典多分辨分析框架构造的小波为第一代小波。

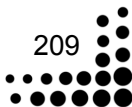
1995 年 Sweldens 提出了一种不依赖于傅里叶变换的新的小波构造方法——提升格式 (lifting scheme)，称为第二代小波变换。提升格式保持了第一代小波的特性，同时又克服了其平移和伸缩的不变性，并具有许多优点，具体表现在以下方面。

(1) 能够用于构造第一代小波，可根据需要来设计小波基。例如，可以先选择一个具有特殊尺度函数的一般多分辨分析，然后利用提升技术来修正该多分辨分析，直到具备设计者所希望性质。这种方法常用于提升小波的消失矩、构造具有插值性质的小波等。应用提升方法构造第一代小波，虽然不能构造全新的小波，但是它使我们认识到小波的构造能够完全在空间域完成。

(2) 能够改进第一代小波变换算法。

- 它提供了一种快速实现方式，与经典的 Mallat 算法相比，运算量减少一半。
- 能够实现小波变换的原位 (in-place) 计算。换言之，整个计算过程不需要申请辅助存储空间，可节省存储单元。
- 逆小波变换的实现非常简单，快速直接而且意义非常明确，可以简单地通过逆序正变换同时交换加减符号得到。
- 很容易实现整数小波变换，可以使小波变换用于信号的无损压缩。
- 能够很容易地处理边界问题。

(3) 可用于第二代小波的构造。这种小波构造方法完全摆脱了傅里叶变换，放弃了二进平移和伸缩的条件，但获得的小波具有第一代小波的所有优点，如时频局部化性质和快速变





换算法。基于此，我们可以在非平移不变区域（如有界区域或曲面）上构造小波基。

## 14.1 小波分解与重构的多相位表示

由于正交小波变换是双正交小波变换的特殊情况，因此，下面仅介绍具有有限脉冲响应（Finite Impulse Response, FIR）的双正交小波滤波器的情况。

设  $\tilde{h}$ 、 $\tilde{g}$ 、 $h$ 、 $g$  是双正交小波滤波器组，它们对应的二通道 Mallat 算法等价的  $z$  变换表示如图 14-1 所示。

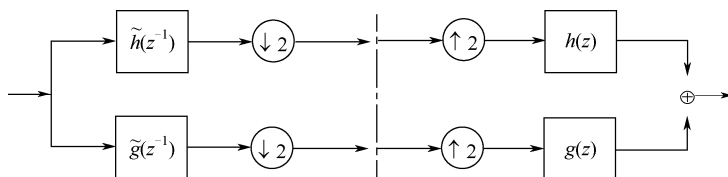


图 14-1 二通道 Mallat 算法等价的  $z$  变换表示

滤波器  $h$  的多相位表示为

$$h(z) = h_e(z^2) + z^{-1}h_o(z^2)$$

其中， $h_e(z)$  包含了  $h(z)$  的偶系数；而  $h_o(z)$  包含了  $h(z)$  的奇系数，即

$$h_e(z) = \sum_k h_{2k} z^{-k}, h_o(z) = \sum_k h_{2k+1} z^{-k}$$

或

$$h_e(z^2) = \frac{h(z) + h(-z)}{2}, h_o(z) = \frac{h(z) - h(-z)}{2z^{-1}}$$

在二通道双正交小波滤波器组的完全重构（Perfect Reconstruction, PR）条件，即

$$\begin{cases} \tilde{h}(z^{-1})h(z) + \tilde{g}(z^{-1})g(z) = 2 \\ h(z)\tilde{h}(-z^{-1}) + g(z)\tilde{g}(-z^{-1}) = 0 \end{cases}$$

下，取  $\tilde{g}(z) = z^{-1}h(-z^{-1})$ ,  $g(z) = z^{-1}\tilde{h}(-z^{-1})$ ，从而有

$$\tilde{h}_e(z) = g_o(z^{-1}), \tilde{h}_o(z) = -g_e(z^{-1}), \tilde{g}_e(z) = -h_o(z^{-1}), \tilde{g}_o(z) = h_e(z^{-1})$$

定义  $h$  和  $g$  的多相位矩阵（polyphase matrix）为

$$\mathbf{P}(z) = \begin{pmatrix} h_e(z) & g_e(z) \\ h_o(z) & g_o(z) \end{pmatrix}$$

从而有

$$\mathbf{P}(z^2)^T = 1/2 \begin{pmatrix} h(z) & h(-z) \\ g(z) & g(-z) \end{pmatrix} \begin{pmatrix} 1 & z \\ 1 & -z \end{pmatrix}$$

类似地，可以定义  $\tilde{h}$  和  $\tilde{g}$  的多相位矩阵，即  $h$  和  $g$  的对偶多相位矩阵为

$$\tilde{\mathbf{P}}(z) = \begin{pmatrix} \tilde{h}_e(z) & \tilde{g}_e(z) \\ \tilde{h}_o(z) & \tilde{g}_o(z) \end{pmatrix}$$

同样地，有



$$\tilde{\mathbf{P}}(z^2)^T = 1/2 \begin{pmatrix} \tilde{h}(z) & \tilde{h}(-z) \\ \tilde{g}(z) & \tilde{g}(-z) \end{pmatrix} \begin{pmatrix} 1 & z \\ 1 & -z \end{pmatrix}$$

因此，小波滤波器的完全重构条件可以写成，

$$\mathbf{P}(z)\tilde{\mathbf{P}}(z^{-1})^T = \mathbf{I} \quad (14-1)$$

其中， $\tilde{\mathbf{P}}(z^{-1})^T$  表示矩阵  $\tilde{\mathbf{P}}(z^{-1})$  的转置矩阵； $\mathbf{I}$  为  $2 \times 2$  的单位矩阵。根据式 (14-1) 可以给出小波分解与重构的多相位表示，如图 14-2 所示。

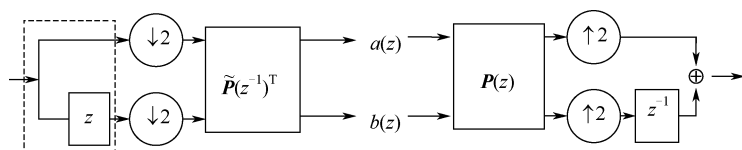


图 14-2 小波分解与重构的多相位表示

下面来说明图 14-1 和图 14-2 在效果上的等价性。设输入序列  $\{x_n\}$  的  $z$  变换为  $x(z)$ ，则图 14-2 中虚线框部分的作用相当于对序列  $\{x_n\}$  进行懒小波变换，即抽取序列  $\{x_n\}$  的偶序列和奇序列。偶序列和奇序列的  $z$  变换分别为

$$e(z) = \frac{1}{2} [x(z^{1/2}) + x(-z^{1/2})] \quad , \quad o(z) = \frac{1}{2} [z^{1/2}x(z^{1/2}) - z^{1/2}x(-z^{1/2})]$$

设这两个  $z$  变换经过  $\tilde{\mathbf{P}}(z^{-1})^T$  作用后的低频部分和高频部分分别为  $a(z)$  和  $d(z)$ ，则有

$$\tilde{\mathbf{P}}(z^{-1})^T \begin{pmatrix} e(z) \\ o(z) \end{pmatrix} = \begin{pmatrix} a(z) \\ d(z) \end{pmatrix}$$

计算上式可以得到

$$a(z) = \frac{1}{2} [\tilde{h}(z^{-1/2})x(z^{1/2}) + \tilde{h}(-z^{-1/2})x(-z^{1/2})]$$

$$d(z) = \frac{1}{2} [\tilde{g}(z^{-1/2})x(z^{1/2}) + \tilde{g}(-z^{-1/2})x(-z^{1/2})]$$

这与根据图 14-1 所得到的结果完全相同。这就说明两图在效果上是完全等价的。

由于  $h$ 、 $g$ 、 $\tilde{h}$ 、 $\tilde{g}$  都是有限长的小波滤波器，所以  $\mathbf{P}(z)$  和  $\tilde{\mathbf{P}}(z)$  的行列式  $\det[\mathbf{P}(z)]$  和  $\det[\tilde{\mathbf{P}}(z)]$  都是 Laurent 多项式。由式 (14-1) 可知， $\det[\mathbf{P}(z)]$  及其倒数都是 Laurent 多项式，故其必定为关于  $z$  的单项式。这里设  $\det[\mathbf{P}(z)] = 1$ 。下面将根据小波分解和重构的多相位表示，通过对多相位矩阵  $\mathbf{P}(z)$  进行因子分解，给出小波变换的提升实现算法。

## 14.2 Laurent 多项式 Euclidean 算法

考虑一个具有紧支集  $[k_b, k_e]$  的有限脉冲响应 (Finite Impulse Response, FIR) 滤波器  $h = \{h_k : k \in \mathbb{Z}\}$ ，其中， $h_k$  对于有限范围  $k_b \leq k \leq k_e$  以外的  $k$  均为零。有限滤波器  $h$  的  $z$  变换是一个 Laurent 多项式  $h(z)$ ，即



$$h(z) = \sum_{k=k_b}^{k_e} h_k z^{-k} \quad (14-2)$$

于是,  $h(z)$  的次数为  $|h(z)| = k_e - k_b$ , 比滤波器的长度  $k_e - k_b + 1$  少 1。当  $h(z)$  是单项式时, 定义其次数为  $-$ 。

**定义 1** 两个 Laurent 多项式的带余除法: 对任何两个 Laurent 多项式  $a(z)$  和  $b(z)$ , 其中,  $b(z) \neq 0, |a(z)| \geq |b(z)|$ , 一定存在 Laurent 多项式  $q(z)$  (称为商) 和  $r(z)$  (称为余数), 使  $a(z) = b(z)q(z) + r(z)$  成立。其中,  $|q(z)| = |a(z)| - |b(z)|, |r(z)| < |b(z)|$  或  $r(z) = 0$ 。称该过程为两个 Laurent 多项式的带余除法, 简称带余除法。

两个 Laurent 多项式的商和余数不是唯一的。例如, 对于  $a(z) = z^{-1} + 6 + z$ ,  $b(z) = 4 + 4z$ , 在以下几种情况下:

$$(1) \quad q(z) = 1/4(z^{-1} + 5), r(z) = -4z;$$

$$(2) \quad q(z) = 1/4(z^{-1} + 1), r(z) = 4;$$

$$(3) \quad q(z) = 1/4(5z^{-1} + 1), r(z) = -4z^{-1}。$$

都满足  $a(z) = b(z)q(z) + r(z)$ , 且  $|q(z)| = |a(z)| - |b(z)|, |r(z)| < |b(z)|$ 。

利用带余除法, 可以给出 Laurent 多项式的 Euclidean 算法: 对于任何两个 Laurent 多项式  $a(z)$  和  $b(z)$ , 其中  $b(z) \neq 0, |a(z)| \geq |b(z)|$ , 设  $a_0(z) = a(z), b_0(z) = b(z)$ , 从  $i = 0$  开始进行如下的递归运算:

$$\begin{aligned} a_{i+1}(z) &= b_i(z) \\ b_{i+1}(z) &= a_i(z) \% b_i(z) \end{aligned}$$

其中, “%” 表示取余运算。则  $a_n(z) = \gcd[a(z), b(z)]$ , 且  $a_n(z)$  是一个 Laurent 多项式, 其中  $n$  为使  $b_n(z) = 0$  的最小数;  $\gcd()$  表示取最大公因子 (greatest common divisor)。

假设  $|b_{i+1}(z)| < |b_i(z)|$ , 则存在  $m$  使得  $|b_m(z)| = 0$ , 因此算法在  $n = m + 1$  步结束, 其中  $n \leq |b(z)| + 1$ 。若记  $q_{i+1}(z) = a_i(z) / b_i(z)$ , 其中, “/” 表示取“商”运算, 则有

$$\begin{pmatrix} a_n(z) \\ 0 \end{pmatrix} = \prod_{i=n}^1 \begin{pmatrix} 0 & 1 \\ 1 & -q_i(z) \end{pmatrix} \begin{pmatrix} a(z) \\ b(z) \end{pmatrix}$$

这等价于

$$\begin{pmatrix} a(z) \\ b(z) \end{pmatrix} = \prod_{i=1}^n \begin{pmatrix} q_i(z) & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} a_n(z) \\ 0 \end{pmatrix} \quad (14-3)$$

显然,  $a_n(z)$  同时整除  $a(z)$  和  $b(z)$ 。如果  $a_n(z)$  是一个单项式, 则  $a(z)$  和  $b(z)$  是互素的。

### 14.3 改进的 Laurent 多项式 Euclidean 算法

根据上节的两个 Laurent 多项式的带余除法的定义, 给出商为单项式的带余除法的定义。

**定义 2** 商为单项式的两个 Laurent 多项式的带余除法: 对任何两个 Laurent 多项式

$$a(z) = \sum_{k=p_a}^{q_a} a_k z^{-k}, \quad p_a, q_a \in \mathbb{Z} \text{ 且 } p_a < q_a$$



$$b(z) = \sum_{k=p_b}^{q_b} b_k z^{-k}, \quad p_b, q_b \in \mathbb{Z} \text{ 且 } p_b < q_b$$

其中,  $b(z) \neq 0, |a(z)| \geq |b(z)|$  (即  $|q_a - p_a| \geq |q_b - p_b|$ ), 一定存在 Laurent 单项式

$$q(z) = \begin{cases} \frac{a_{p_a}}{b_{p_b}} z^{p_b - p_a}, & p = \min(p_a, p_b) < 0 \\ \frac{a_{q_a}}{b_{q_b}} z^{q_b - q_a}, & p = \min(p_a, p_b) \geq 0 \end{cases} \quad (14-4)$$

称为单项商, 其和多项式  $r(z)$  (称为余数), 使  $a(z) = b(z)q(z) + r(z)$  成立。称该过程为商为单项式的两个 Laurent 多项式的带余除法, 简称单项商带余除法。

从上面的定义可以看出, 单项商带余除法的定义少了一个约束条件  $|r(z)| < |b(z)|$ 。这是因为, 在要求商为单项式的条件下, 就不能保证所求得的余数  $r(z)$  的次数一定比除数  $b(z)$  的小。另外, 在定义中也可以不要求  $|a(z)| \geq |b(z)|$ , 为与前面所述的 Euclidean 算法相一致, 这里特意增加了该条件。

由于对于给定的两个 Laurent 多项式  $a(z)$  和  $b(z)$ ,  $p_a, q_a, p_b, q_b$  以及  $a_{p_a}, b_{p_b}, a_{q_a}, b_{q_b}$  都是确定的数, 因此在该定义下, 显然有下面的定理成立。

定理 1 两个 Laurent 多项式的单项商和余数是唯一的。

例如, 对于上面的例子,  $a(z) = z^{-1} + 6 + z$ ,  $b(z) = 4 + 4z$ 。

由于  $p = \min(p_a, p_b) = \min(-1, 0) = -1 < 0$ , 故  $q(z) = z^{-1} / 4$ ,  $r(z) = 5 + z$ , 且  $|r(z)| = |b(z)| = 1$ 。

利用单项商带余除法, 我们可以给出 Laurent 多项式的改进的 Euclidean 算法。

对于任何两个 Laurent 多项式  $a(z)$  和  $b(z)$ , 其中  $b(z) \neq 0, |a(z)| \geq |b(z)|$ 。

(1) 初始化, 令  $i = 0$ ,  $a_0(z) = a(z), b_0(z) = b(z)$ 。

(2) 计算  $a_{i+1}(z) = b_i(z)$ 。

(3) 计算单项商

$$q_{i+1}(z) = \begin{cases} \frac{a_{p_a}^i}{b_{p_b}^i} z^{p_b - p_a}, & p = \min(p_a, p_b) < 0 \\ \frac{a_{q_a}^i}{b_{q_b}^i} z^{q_b - q_a}, & p = \min(p_a, p_b) \geq 0 \end{cases}$$

其中,  $a_{p_a}^i, a_{q_a}^i$  和  $b_{p_b}^i, b_{q_b}^i$  分别表示  $a_i(z)$  和  $b_i(z)$  中  $z$  的最大指数项及最小指数项的系数。

(4) 计算余数

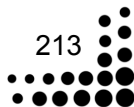
$$b_{i+1}(z) = a_i(z) \% b_i(z) = a_i(z) - q_{i+1}(z)b_i(z)$$

其中, “%” 表示单项商带余除法定义下的取余运算。

(5)  $i = i + 1$ 。

(6) 如果  $b_i(z) = 0$  退出循环; 否则, 跳转到(2)继续循环。

和前面的定义一样,  $a_n(z) = \gcd[a(z), b(z)]$ , 且  $a_n(z)$  是一个 Laurent 多项式, 其中  $n$  为使  $b_n(z) = 0$  的最小数。假设  $|b_{i+1}(z)| < |b_i(z)|$ , 则存在  $m$  使得  $|b_m(z)| = 0$ , 因此算法在  $n = m + 1$  步





结束。其中,  $n = |a(z)| + |b(z)| + |p_a - p_b| + |q_a - q_b|$ 。同样我们可以得到与式(14-3)相同的结果, 即

$$\begin{pmatrix} a(z) \\ b(z) \end{pmatrix} = \prod_{i=1}^n \begin{pmatrix} q_i(z) & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} a_n(z) \\ 0 \end{pmatrix}$$

显然,  $a_n(z)$  同时整除  $a(z)$  和  $b(z)$ 。如果  $a_n(z)$  是一个单项式, 则  $a(z)$  和  $b(z)$  是互素的。

比较改进前后的 Laurent 多项式 Euclidean 算法, 我们可以发现, 虽然两种算法都可以得到与式(14-3)相同的结构, 但是其中的  $q_i(z)$  是完全不同的。前者中的  $q_i(z)$  为一个 Laurent 多项式, 而后者中所有的  $q_i(z)$  都是单项式, 使得各个矩阵之间的乘积变得简单, 计算复杂度大大减少。

下面还是用上面的例子  $a(z) = z^{-1} + 6 + z$ ,  $b(z) = 4 + 4z$  来说明。

**【例 14-1】**用基于带余除法的 Laurent 多项式 Euclidean 算法进行计算。

计算过程 (由于结果不止一种, 这里只取下面一种):

(1) 令  $a_0(z) = a(z) = z^{-1} + 6 + z$ ,  $b_0 = b(z) = 4 + 4z$ ;

(2)  $a_1(z) = b_0(z) = 4 + 4z$ ,  $b_1(z) = a_0 \% b_0 = 4$ ,  $q_1(z) = 1/4z^{-1} + 1/4$ ;

(3)  $a_2(z) = b_1(z) = 4$ ,  $b_2(z) = a_1 \% b_1 = 0$ ,  $q_2(z) = 1 + z$ 。

所以,  $a(z)$  和  $b(z)$  是互素的, 且

$$\begin{pmatrix} z^{-1} + 6 + z \\ 4 + 4z \end{pmatrix} = \begin{pmatrix} 1/4z^{-1} + 1/4 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 + z & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 4 \\ 0 \end{pmatrix}$$

辗转除法的步数为  $n = 2 = |b(z)| + 1$ 。

**【例 14-2】**用基于单项商带余除法的 Laurent 多项式 Euclidean 算法进行计算。计算过程:

(1) 令  $a_0(z) = a(z) = z^{-1} + 6 + z$ ,  $b_0 = b(z) = 4 + 4z$ ;

(2)  $a_1(z) = b_0(z) = 4 + 4z$ ,  $b_1(z) = a_0 \% b_0 = 5 + z$ ,  $q_1(z) = 1/4z^{-1}$ ;

(3)  $a_2(z) = b_1(z) = 5 + z$ ,  $b_2(z) = a_1 \% b_1 = -16$ ,  $q_2(z) = 4$ ;

(4)  $a_3(z) = b_2(z) = -16$ ,  $b_3(z) = a_2 \% b_2 = 5$ ,  $q_3(z) = -1/16z$ ;

(5)  $a_4(z) = b_3(z) = 5$ ,  $b_4(z) = a_3 \% b_3 = 0$ ,  $q_4(z) = -16/5$ 。

所以,  $a(z)$  和  $b(z)$  是互素的, 且

$$\begin{pmatrix} z^{-1} + 6 + z \\ 4 + 4z \end{pmatrix} = \begin{pmatrix} 1/4z^{-1} & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 4 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} -1/16z & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} -16/5 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 5 \\ 0 \end{pmatrix}$$

辗转除法的步数为  $n = 4 = |a(z)| + |b(z)| + |-1 - 0| + |1 - 1|$ 。

下面用  $a(z) = z^{-2} + 6 + z^2$ ,  $b(z) = 4 + 4z$  来说明。

**【例 14-3】**用基于带余除法的 Laurent 多项式 Euclidean 算法进行计算。

计算过程:

(1) 令  $a_0(z) = a(z) = z^{-2} + 6 + z^2$ ,  $b_0 = b(z) = 4 + 4z$ ;

(2)  $a_1(z) = b_0(z) = 4 + 4z$ ,  $b_1(z) = a_0 \% b_0 = 8z^{-2}$ ,  $q_1(z) = 1/4z^{-3} + 1/4z^2 + 7/4z - 7/4$ ;

(3)  $a_2(z) = b_1(z) = 8z^{-2}$ ,  $b_2(z) = a_1 \% b_1 = 0$ ,  $q_2(z) = 1/2z(1 + z)$ 。

所以,  $a(z)$  和  $b(z)$  是互素的, 且



$$\begin{pmatrix} z^{-2} + 6 + z^2 \\ 4 + 4z \end{pmatrix} = \begin{pmatrix} 1/4z^{-3} + 1/4z^2 + 7/4z - 7/4 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1/2z(1+z) & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 8z^{-2} \\ 0 \end{pmatrix}$$

辗转除法的步数为  $n = 2 = |b(z)| + 1$ 。

【例 14-4】用基于单项商带余除法的 Laurent 多项式 Euclidean 算法进行计算。

计算过程：

- (1) 令  $a_0(z) = a(z) = z^{-2} + 6 + z^2$ ,  $b_0 = b(z) = 4 + 4z$ ;
- (2)  $a_1(z) = b_0(z) = 4 + 4z$ ,  $b_1(z) = a_0 \% b_0 = 6 - z^{-1} + z^2$ ,  $q_1(z) = 1/4z^{-2}$ ;
- (3)  $a_2(z) = b_1(z) = 6 - z^{-1} + z^2$ ,  $b_2(z) = a_1 \% b_1 = 28z + 4z^3$ ,  $q_2(z) = -4z$ ;
- (4)  $a_3(z) = b_2(z) = 28z + 4z^3$ ,  $b_3(z) = a_2 \% b_2 = 6 + 1/7z + z^2$ ,  $q_3(z) = -1/28z^{-2}$ ;
- (5)  $a_4(z) = b_3(z) = 6 + 1/7z + z^2$ ,  $b_4(z) = a_3 \% b_3 = -4/7z^2 + 4z$ ,  $q_4(z) = 4z$ ;
- (6)  $a_5(z) = b_4(z) = -4/7z^2 + 4z$ ,  $b_5(z) = a_4 \% b_4 = 50/7z + 6$ ,  $q_5(z) = -7/4$ ;
- (7)  $a_6(z) = b_5(z) = 50/7z + 6$ ,  $b_6(z) = a_5 \% b_5 = 112/25z$ ,  $q_6(z) = -4/50z$ ;
- (8)  $a_7(z) = b_6(z) = 112/25z$ ,  $b_7(z) = a_6 \% b_6 = 6$ ,  $q_7(z) = 625/392$ ;
- (9)  $a_8(z) = b_7(z) = 625/392$ ,  $b_8(z) = a_7 \% b_7 = 0$ ,  $q_8(z) = 56/75z$ 。

所以,  $a(z)$  和  $b(z)$  是互素的, 且

$$\begin{pmatrix} z^{-2} + 6 + z^2 \\ 4 + 4z \end{pmatrix} = \begin{pmatrix} 1/4z^{-2} & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} -4z & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} -1/28z^{-2} & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 4z & 1 \\ 1 & 0 \end{pmatrix} \times \\ \begin{pmatrix} -7/4 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} -4/50z & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 625/392 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 56/75z & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 5 \\ 0 \end{pmatrix}$$

辗转除法的步数为  $n = 8 = |a(z)| + |b(z)| + |-2 - 0| + |2 - 1|$ 。

## 14.4 多相位矩阵的因子分解

Daubechies 和 Swedens 在其文献中研究并给出了多相位矩阵因子分解的定理, 该定理构成了小波变换提升实现的基础。下面在  $P(z)$  的行列式等于 1 的前提下, 讨论完全重构滤波器的提升分解格式。并给出多相位矩阵的因子分解定理。

定义 3 若多相位矩阵  $P(z)$  的行列式等于 1, 则相应的滤波器对  $(h, g)$  称为补。

显然, 当  $(h, g)$  构成补时,  $(\tilde{h}, \tilde{g})$  也构成补。

定理 2 (提升) 设  $(h, g)$  构成补, 则任何其他  $h$  的补  $g^{\text{new}}(z)$  都可以表示为

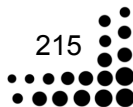
$$g^{\text{new}}(z) = g(z) + h(z)s(z^2)$$

其中,  $s(z^2)$  为 Laurent 多项式。

证明: 直接计算, 知道  $h(z)s(z^2)$  的多相位分量对于偶数部分为  $h_e(z)s(z)$ , 而对于奇数部分为  $h_o(z)s(z)$ 。做一次提升分解后, 新的多相位分解矩阵定义为

$$P^{\text{new}}(z) = P(z) \begin{pmatrix} 1 & s(z) \\ 0 & 1 \end{pmatrix}$$

此时, 行列式的值没有发生变化, 而  $g^{\text{new}}(z) = g(z) + h(z)s(z^2)$ 。类似地, 可以得到对偶







多相位分解矩阵为

$$\tilde{\mathbf{P}}^{\text{new}}(z) = \tilde{\mathbf{P}}(z) \begin{pmatrix} 1 & 0 \\ -s(z^{-1}) & 1 \end{pmatrix}$$

此时，提升格式得到新的滤波器  $\tilde{h}^{\text{new}}(z) = \tilde{h}(z) - \tilde{g}(z)s(z^{-2})$ 。

类似地，我们也可以通过偶提升格式来建立分解与重构端的提升结构。

定理 3（对偶提升） 设  $(h, g)$  构成补，则任何其他  $g$  的补  $h^{\text{new}}(z)$  都可以表示为

$$h^{\text{new}}(z) = h(z) - g(z)t(z^2)$$

其中， $t(z)$  为 Laurent 多项式。

做一次对偶提升分解后，新的多相位分解矩阵定义为

$$\mathbf{P}^{\text{new}}(z) = \mathbf{P}(z) \begin{pmatrix} 1 & 0 \\ t(z) & 1 \end{pmatrix}$$

而对应于  $\tilde{g}$  产生新的滤波器  $\tilde{g}^{\text{new}}$ ，满足  $\tilde{g}^{\text{new}}(z) = \tilde{g}(z) - \tilde{h}(z)t(z^{-2})$ 。

根据前面所述的改进的 Euclidean 算法，对多项式向量

$$\begin{pmatrix} h_e(z) \\ h_o(z) \end{pmatrix} \quad (14-5)$$

进行分解，可以得到

$$\begin{pmatrix} h_e(z) \\ h_o(z) \end{pmatrix} = \prod_{i=1}^n \begin{pmatrix} q_i(z) & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} K \\ 0 \end{pmatrix}$$

经过处理，我们总能保证  $n$  为偶数。当  $n$  为奇数时，经过下面的处理过程可以使得  $n$  变为偶数。

假设  $n$  为奇数，并令

$$a_{n-1}(z) = A(z), b_{n-1}(z) = K$$

$$a_n(z) = b_{n-1}(z) = K, b_n(z) = a_{n-1}(z) \% b_{n-1}(z) = 0, q_n(z) = Q(z)$$

则

$$a_{n-1}(z) = q_n(z)b_{n-1}(z) = KQ(z)$$

现在，如果令  $q_n(z) = Q(z)/2$ ，则

$$a_n(z) = b_{n-1}(z) = B(z), b_n(z) = a_{n-1}(z) \% b_{n-1}(z) = a_{n-1}(z)/2 = KQ(z)/2$$

$$a_{n+1}(z) = b_n(z) = KQ(z)/2, b_{n+1}(z) = a_n(z) \% b_n(z) = 0, q_{n+1}(z) = 2/Q(z)$$

由于  $Q(z)$  为单项式，所以  $q_{n+1}(z)$  仍为单项式，则此时有

$$\begin{pmatrix} h_e(z) \\ h_o(z) \end{pmatrix} = \prod_{i=1}^{n+1} \begin{pmatrix} q_i(z) & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} Kq_n(z)/2 \\ 0 \end{pmatrix} \quad (14-6)$$

其中， $q_n(z) = q_n(z)/2$ ;  $q_{n+1}(z) = 2/q_n(z)$ 。

给定一个滤波器  $h$ ，通常可以用下面的式子得到与其构成补的滤波器  $g^0$ ：

$$\mathbf{P}^0(z) = \begin{pmatrix} h_e(z) & g_e^0(z) \\ h_o(z) & g_o^0(z) \end{pmatrix} = \prod_{i=1}^n \begin{pmatrix} q_i(z) & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} K & 0 \\ 0 & 1/K \end{pmatrix}$$

根据前面的论述，上式构成了一种完全重构滤波器多相位矩阵的提升分解。但是，对于一般给定的特殊长度（包括分解和重构）的完全重构滤波器  $(h, g)$ ，上述分解不一定满足。因此，我们需要根据定理 2 对该结果进行提升。存在 Laurent 多项式  $s(z)$ ，使得



$g(z) = g^0(z) + h(z)s(z^2)$ , 则滤波器对  $(h, g)$  的多相位矩阵  $P(z)$  可以表示为

$$P(z) = \begin{pmatrix} h_e(z) & g_e(z) \\ h_o(z) & g_o(z) \end{pmatrix} = P^0(z) \begin{pmatrix} 1 & s(z) \\ 0 & 1 \end{pmatrix} \quad (14-7)$$

另外, 注意到

$$\begin{pmatrix} q_i(z) & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & q_i(z) \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ q_i(z) & 1 \end{pmatrix}$$

当  $n$  为偶数时, 利用上式可以得到

$$P^0(z) = \prod_{i=1}^{n/2} \begin{pmatrix} 1 & q_{2i-1}(z) \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ q_{2i}(z) & 1 \end{pmatrix} \begin{pmatrix} K & 0 \\ 0 & 1/K \end{pmatrix} \quad (14-8)$$

根据上面的讨论, 我们可以给出下面的定理。

**定理 4** 若  $P(z)$  的行列式等于 1, 即  $\det(P(z)) = 1$ , 则总存在 Laurent 多项式  $u_i(z)$  和  $p_i(z)$  ( $1 \leq i \leq m$ ) 及非零常数  $K$ , 使得

$$P(z) = \prod_{i=1}^m \begin{pmatrix} 1 & u_i(z) \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ p_i(z) & 1 \end{pmatrix} \begin{pmatrix} K & 0 \\ 0 & 1/K \end{pmatrix}$$

其中,  $p_m(z) = 0$ ;  $u_m(z) = K^2 s(z)$ 。

在式 (14-8) 中, 令  $m = n/2 + 1$ ,  $p_m(z) = 0$ ,  $u_m(z) = K^2 s(z)$ , 并结合式 (14-7), 可以知道该定理成立。

下面给出提升因子  $u_i(z)$  和  $p_i(z)$  ( $1 \leq i \leq m$ ) 的计算方法。

首先, 式 (14-5) 应用欧几里得算法, 可得到

$$\begin{pmatrix} h_e(z) \\ h_o(z) \end{pmatrix} = \prod_{i=1}^{m-1} \begin{pmatrix} 1 & u_i(z) \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ p_i(z) & 1 \end{pmatrix} \begin{pmatrix} K \\ 0 \end{pmatrix}$$

$$\text{令 } P^0(z) = \begin{pmatrix} h_e(z) & g_e^0(z) \\ h_o(z) & g_o^0(z) \end{pmatrix} = \prod_{i=1}^{m-1} \begin{pmatrix} 1 & u_i(z) \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ p_i(z) & 1 \end{pmatrix} \begin{pmatrix} K & 0 \\ 0 & 1/K \end{pmatrix}$$

注意到

$$\begin{pmatrix} 1 & K^2 s(z) \\ 0 & 1 \end{pmatrix} \begin{pmatrix} K & 0 \\ 0 & 1/K \end{pmatrix} = \begin{pmatrix} K & Ks(z) \\ 0 & 1/K \end{pmatrix} = \begin{pmatrix} K & 0 \\ 0 & 1/K \end{pmatrix} \begin{pmatrix} 1 & s(z) \\ 0 & 1 \end{pmatrix}$$

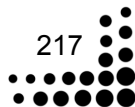
则由式 (14-7) 可得,  $P(z) = P^0(z) \begin{pmatrix} 1 & s(z) \\ 0 & 1 \end{pmatrix}$  其中,  $s(z) = u_m(z) / K^2$ 。

若记  $Q(z) = \prod_{i=1}^{m-1} \begin{pmatrix} 1 & u_i(z) \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ p_i(z) & 1 \end{pmatrix} = \begin{pmatrix} a(z) & c(z) \\ b(z) & d(z) \end{pmatrix}$ , 则

$$P^0(z) = \begin{pmatrix} h_e(z) & g_e^0(z) \\ h_o(z) & g_o^0(z) \end{pmatrix} = Q(z) \begin{pmatrix} K & 0 \\ 0 & 1/K \end{pmatrix} = \begin{pmatrix} Ka(z) & c(z)/K \\ Kb(z) & d(z)/K \end{pmatrix}$$

则有

$$\begin{pmatrix} 1 & s(z) \\ 0 & 1 \end{pmatrix} = (P^0(z))^{-1} P(z) = \begin{pmatrix} Ka(z) & c(z)/K \\ Kb(z) & d(z)/K \end{pmatrix}^{-1} \begin{pmatrix} h_e(z) & g_e(z) \\ h_o(z) & g_o(z) \end{pmatrix}$$





$$= \begin{pmatrix} \frac{d(z)}{K} h_e(z) - \frac{c(z)}{K} h_o(z) & \frac{d(z)}{K} g_e(z) - \frac{c(z)}{K} g_o(z) \\ -Kb(z)h_e(z) + Kh_o(z)a(z) & -Kb(z)g_e(z) + Kg_o(z)a(z) \end{pmatrix}$$

因此有

$$s(z) = \frac{d(z)g_e(z) - c(z)g_o(z)}{K} = g_o^0(z)g_e(z) - g_e^0(z)g_o(z) = \begin{vmatrix} g_e(z) & g_e^0(z) \\ g_o(z) & g_o^0(z) \end{vmatrix},$$

$$u_m(z) = K^2 s(z) = K^2 \begin{vmatrix} g_e(z) & g_e^0(z) \\ g_o(z) & g_o^0(z) \end{vmatrix}$$

综上所述, 我们可以得到有限滤波器多相位矩阵的提升分解算法。

(1) 使用欧几里得算法得到

$$\begin{pmatrix} h_e(z) \\ h_o(z) \end{pmatrix} = \prod_{i=1}^{m-1} \begin{pmatrix} 1 & u_i(z) \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ p_i(z) & 1 \end{pmatrix} \begin{pmatrix} K \\ 0 \end{pmatrix}$$

(2) 计算

$$\mathbf{P}^0(z) = \begin{pmatrix} h_e(z) & g_e^0(z) \\ h_o(z) & g_o^0(z) \end{pmatrix} = \prod_{i=1}^{m-1} \begin{pmatrix} 1 & u_i(z) \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ p_i(z) & 1 \end{pmatrix} \begin{pmatrix} K & 0 \\ 0 & 1/K \end{pmatrix}$$

(3) 最后计算出  $u_m(z)$

$$u_m(z) = K^2 s(z) = K^2 \begin{vmatrix} g_e(z) & g_e^0(z) \\ g_o(z) & g_o^0(z) \end{vmatrix}$$

另外, 对式 (14-5) 应用欧几里得算法, 可得到

$$\begin{pmatrix} h_e(z) \\ h_o(z) \end{pmatrix} = \prod_{i=1}^n \begin{pmatrix} q_i(z) & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} K \\ 0 \end{pmatrix}$$

其中,  $n$  为偶数, 若为奇数, 可以根据式 (14-6) 处理, 最终可以使得  $n$  为偶数。

令

$$\begin{pmatrix} a(z) & b(z) \\ c(z) & d(z) \end{pmatrix} = \prod_{i=1}^{n-1} \begin{pmatrix} q_i(z) & 1 \\ 1 & 0 \end{pmatrix}$$

则

$$\mathbf{P}^0(z) = \begin{pmatrix} h_e(z) & g_e^0(z) \\ h_o(z) & g_o^0(z) \end{pmatrix} = \begin{pmatrix} a(z) & b(z) \\ c(z) & d(z) \end{pmatrix} \begin{pmatrix} q_n(z) & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} K & 0 \\ 0 & 1/K \end{pmatrix}$$

化简上式, 可以得到

$$g_e^0(z) = a(z)/K, g_o^0(z) = c(z)/K$$

则根据式 (14-7) 可以得到

$$u_m(z) = K^2 s(z) = Kc(z)g_e(z) - Ka(z)g_o(z)$$

用上式进行计算, 比前述算法少做一次矩阵乘法。

根据二通道小波滤波器的完全重构 (perfect reconstruction) 条件的多相位表示

$$\mathbf{P}(z)\tilde{\mathbf{P}}(z^{-1})^T = \mathbf{I} \quad (14-9)$$

式中,  $\mathbf{I}$  为  $2 \times 2$  的单位矩阵。可以得到

$$\tilde{\mathbf{P}}(z^{-1})^T = \begin{pmatrix} 1/K & 0 \\ 0 & K \end{pmatrix} \prod_{i=m}^1 \begin{pmatrix} 1 & 0 \\ -p_i(z) & 1 \end{pmatrix} \begin{pmatrix} 1 & -u_i(z) \\ 0 & 1 \end{pmatrix} \quad (14-10)$$

由此, 我们可给出基于提升的正向小波变换和逆向小波变换的流程图分别如图 14-3



和图 14-4 所示。

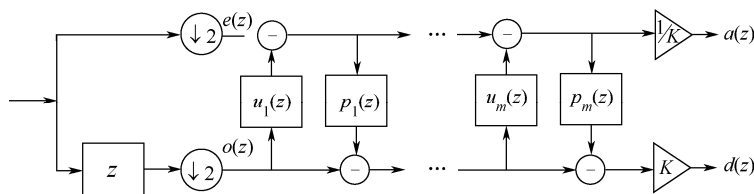


图 14-3 基于提升的正向小波变换流程图

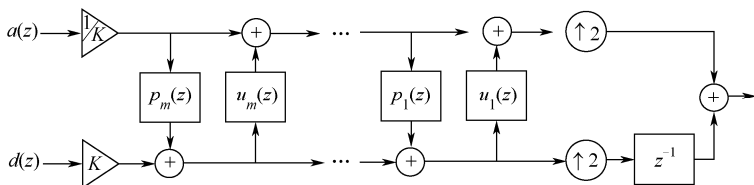


图 14-4 基于提升的逆向小波变换流程图

图 14-3 和图 14-4 中,  $e(z)$  和  $o(z)$  分别表示输入序列的偶序列和奇序列的  $z$  变换;  $a(z)$  和  $d(z)$  分别表示经过一级小波变换后得到的低频部分和高频部分的  $z$  变换。

## 14.5 小波变换的提升实现的传统算法

根据上节的讨论,下面给出小波变换的提升实现算法。根据  $u_1(z)$  是否为零分为两种情况。当  $u_1(z) \neq 0$  时,预测步骤的起始步由奇序列预测偶序列。 $u_1(z) = 0$  时,预测步骤的起始步由偶序列预测奇序列。限于篇幅,这里只给出  $u_1(z) \neq 0$  的情况。

设  $x = \{x_0, x_1, \dots, x_{N-1}\}$  是长度为  $N$  (这里设  $N$  为偶数) 的一个信号,  $s^0$  和  $d^0$  表示它的偶序列和奇序列。由于  $u_i(z)$  和  $p_i(z)$  都是单项式,故可以记

$$u_i(z) = \sum_k u_k^i z^{-k}, \quad p_i(z) = \sum_k p_k^i z^{-k} \quad (i=1, 2, \dots, m)$$

若记  $s_i(z)$ 、 $d_i(z)$  分别表示  $s^i = \{s_l^i\}$ 、 $d^i = \{d_l^i\}$  ( $i=1, \dots, m$ ) 的  $z$  变换,且

$$\begin{pmatrix} s_i(z) \\ d_i(z) \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ -p_i(z) & 1 \end{pmatrix} \begin{pmatrix} 1 & -u_i(z) \\ 0 & 1 \end{pmatrix} \begin{pmatrix} s_{i-1}(z) \\ d_{i-1}(z) \end{pmatrix}$$

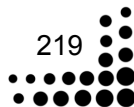
则

$$\begin{cases} s_i(z) = s_{i-1}(z) - u_i(z)d_{i-1}(z) \\ d_i(z) = d_{i-1}(z) - p_i(z)s_{i-1}(z) \end{cases}, \quad i=1, 2, \dots, m$$

用序列卷积可表示为

$$\begin{cases} s_l^i = s_l^{i-1} - (u^i * d^{i-1})_l = s_l^{i-1} - \sum_k u_k^i d_{l-k}^{i-1} \\ d_l^i = d_l^{i-1} - (p^i * s^i)_l = d_l^{i-1} - \sum_k p_k^i s_{l-k}^i \end{cases}, \quad i=1, 2, \dots, m$$

故可以写出正向小波变换提升实现的简化算法如下。





(1) 懒小波变换 (Lazy wavelet transform):

$$s_l^0 = x_{2l}, d_l^0 = x_{2l+1}, l = 0, 1, \dots, N/2 - 1$$

(2)  $m$  步提升及对偶提升 ( $m$  lifting and dual lifting steps):

$$\begin{cases} s_l^i = s_l^{i-1} - u_k^i d_{l-k}^{i-1} \\ d_l^i = d_l^{i-1} - p_k^i s_{l-k}^{i-1} \end{cases}, l = 0, 1, \dots, N/2 - 1; i = 1, 2, \dots, m$$

(3) 比例缩放变换 (scaling):

$$\begin{cases} s_l = s_l^m / K \\ d_l = K d_l^m \end{cases}, l = 0, 1, \dots, N/2 - 1$$

最后得到的  $s$  和  $d$  分别为小波分解的低频分量和高频分量。其中,  $s = \{s_0, s_1, \dots, s_{N/2-1}\}$ ,  $d = \{d_0, d_1, \dots, d_{N/2-1}\}$ 。通过对正向小波变换按照相反的步骤操作, 同时改变正负号, 即可得到相应的逆变换。

(1) 比例缩放变换 (scaling):

$$\begin{cases} s_l = s_l^m / K \\ d_l = K d_l^m \end{cases}, l = 0, 1, \dots, N/2 - 1$$

(2)  $m$  步提升及对偶提升 ( $m$  lifting and dual lifting steps):

$$\begin{cases} d_l^{i-1} = d_l^i + p_k^i s_{l-k}^i \\ s_l^{i-1} = s_l^i + u_k^i d_{l-k}^{i-1} \end{cases}, l = 0, 1, \dots, N/2 - 1; i = m, m-1, \dots, 2, 1$$

(3) 逆懒小波变换 (inverse Lazy wavelet transform)

$$x_{2l} = s_l^0, x_{2l+1} = d_l^0, l = 0, 1, \dots, N/2 - 1$$

另外, 参照上面的介绍,  $u_1(z) = 0$  时的算法步骤可以类似给出。这里不再列出。

## 14.6 小波变换的提升实现的简化算法

利用上面给出的改进的 Laurent 多项式的 Euclidean 算法, 我们提出一种小波变换的提升实现的简化算法。该算法与前面的算法类似, 只是提升与对偶提升步骤稍有差别, 其他过程都基本一致。同样, 该算法也按  $u_1(z)$  是否为零分为两种情况。  $u_1(z) \neq 0$  时的算法过程如下。

设  $x = \{x_0, x_1, \dots, x_{N-1}\}$  是长度为  $N$  (这里设  $N$  为偶数) 的一个信号,  $s^0$  和  $d^0$  表示它的偶序列和奇序列。由于  $u_i(z)$  和  $p_i(z)$  都是单项式, 故可以记

$$u_i(z) = u^i z^{-u_i}, p_i(z) = p^i z^{-p_i} (i = 1, 2, \dots, m-1), u_m(z) = \sum_k u_k^m z^{-k}$$

若记  $s_i(z)$ 、 $d_i(z)$  分别表示  $s^i = \{s_l^i\}$ 、 $d^i = \{d_l^i\}$  ( $i = 1, \dots, m$ ) 的  $z$  变换, 且

$$\begin{pmatrix} s_i(z) \\ d_i(z) \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ -p_i(z) & 1 \end{pmatrix} \begin{pmatrix} 1 & -u_i(z) \\ 0 & 1 \end{pmatrix} \begin{pmatrix} s_{i-1}(z) \\ d_{i-1}(z) \end{pmatrix}$$

则

$$\begin{cases} s_i(z) = s_{i-1}(z) - u_i(z) d_{i-1}(z) \\ d_i(z) = d_{i-1}(z) - p_i(z) s_{i-1}(z) \end{cases}, i = 1, 2, \dots, m-1$$



用序列卷积可表示为

$$\begin{cases} s_l^i = s_l^{i-1} - (u^i * d^{i-1})_l = s_l^{i-1} - u^i d_{l+u_i}^{i-1}, i = 1, 2, \dots, m-1 \\ d_l^i = d_l^{i-1} - (p^i * s^i)_l = d_l^{i-1} - p^i s_{l+p_i}^i \end{cases}$$

$$s_l^m(z) = s_l^{m-1}(z) - (u^m * d^{m-1})_l = s_l^{m-1}(z) - \sum_k u_k^m d_{l-k}^{m-1}$$

故可以写出正向小波变换提升实现的简化算法。

(1) 懒小波变换 (Lazy wavelet):

$$s_l^0 = x_{2l}, d_l^0 = x_{2l+1}, l = 0, 1, \dots, N/2 - 1$$

(2)  $m$  步提升及对偶提升 ( $m$  lifting and dual lifting steps):

$$\begin{cases} s_l^i = s_l^{i-1} - u^i d_{l+u_i}^{i-1}, l = 0, 1, \dots, N/2 - 1; i = 1, 2, \dots, m-1 \\ d_l^i = d_l^{i-1} - p^i s_{l+p_i}^i \end{cases}$$

$$s_l^m(z) = s_l^{m-1}(z) - (u^m * d^{m-1})_l = s_l^{m-1}(z) - \sum_k u_k^m d_{l-k}^{m-1}$$

(3) 比例缩放变换 (scaling):

$$\begin{cases} s_l = s_l^m / K \\ d_l = K d_l^m, l = 0, 1, \dots, N/2 - 1 \end{cases}$$

最后得到的  $s$  和  $d$  分别为小波分解的低频分量和高频分量。其中,  $s = \{s_0, s_1, \dots, s_{N/2-1}\}$ ,  $d = \{d_0, d_1, \dots, d_{N/2-1}\}$ 。通过对正向小波变换按照相反的步骤操作, 同时改变正负号, 即可得到相应的逆变换。另外, 当  $u_1(z) = 0$  的情况与  $u_1(z) \neq 0$  时情况稍有不同, 但是可以参照传统算法  $u_1(z) = 0$  时的情况类似给出算法步骤, 这里不再列出。

## 14.7 提升算法举例

下面以几种常用的小波变换的提升实现为例来说明本节算法与传统算法的不同。由于传统的多相位矩阵的分解不唯一, 所以小波变换的提升分解也不是唯一的, 以下用传统小波变换提升分解算法实现的例子中只给出一种常用的分解形式。

【例 14-5】用传统提升小波算法实现小波变换的提升。

小波滤波器如下:

$$\tilde{h} = \left\{ -\frac{1}{4\sqrt{2}}, \frac{1}{2\sqrt{2}}, \frac{3}{2\sqrt{2}}, \frac{1}{2\sqrt{2}}, -\frac{1}{4\sqrt{2}} \right\}, h = \left\{ \frac{1}{2\sqrt{2}}, \frac{1}{\sqrt{2}}, \frac{1}{2\sqrt{2}} \right\}$$

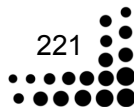
可以求出小波滤波器的多相位矩阵在传统的提升小波算法下存在如下一种因子分解:

$$\tilde{P}(z) = \begin{pmatrix} 1 & \tau(1+z^{-1}) \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \nu(1+z) & 1 \end{pmatrix} \begin{pmatrix} \omega & 0 \\ 0 & 1/\omega \end{pmatrix}$$

其中,  $\tau = -0.5, \nu = 0.25, \omega = \sqrt{2}$ 。

由此可以给出正向小波变换的提升实现为

$$s_l^{(0)} = x_{2l}, d_l^{(0)} = x_{2l+1}$$





$$d_l^{(1)} = d_l^{(0)} + \tau [s_l^{(0)} + s_{l+1}^{(0)}]$$

$$s_l^{(1)} = s_l^{(0)} + \nu [d_l^{(1)} + d_{l-1}^{(1)}]$$

$$s_l = \omega s_l^{(1)}, d_l = d_l^{(1)} / \omega$$

【例 14-6】用提升小波算法实现式(14-3)小波变换的提升。

可以求出,  $\tilde{P}(z)$  因子分解为

$$\tilde{P}(z) = \begin{pmatrix} 1 & \tau z^{-1} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \nu & 1 \end{pmatrix} \begin{pmatrix} 1 & \omega z \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \lambda & 1 \end{pmatrix} \begin{pmatrix} 1 & \delta \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \zeta & 0 \\ 0 & 1/\zeta \end{pmatrix}$$

其中,  $\tau = -0.5, \nu = -2, \omega = -0.0625, \lambda = 2.28571429, \delta = -0.43748, \zeta = 1.23743687$ 。

由  $\tilde{P}(z)$  可以得到

$$\tilde{P}(z^{-1})^T = \begin{pmatrix} \zeta & 0 \\ 0 & 1/\zeta \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \delta & 1 \end{pmatrix} \begin{pmatrix} 1 & \lambda \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \omega z & 1 \end{pmatrix} \begin{pmatrix} 1 & \nu \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \tau z & 1 \end{pmatrix}$$

由此可以给出正向小波变换的提升实现为

$$s_l^{(0)} = x_{2l}, d_l^{(0)} = x_{2l+1}$$

$$d_l^{(1)} = d_l^{(0)} + \tau s_{l+1}^{(0)}$$

$$s_l^{(1)} = s_l^{(0)} + \nu d_l^{(1)}$$

$$d_l^{(2)} = d_l^{(1)} + \omega s_{l+1}^{(1)}$$

$$s_l^{(2)} = s_l^{(1)} + \lambda d_l^{(2)}$$

$$d_l^{(3)} = d_l^{(2)} + \delta s_l^{(2)}$$

$$s_l = \zeta s_l^{(3)}, d_l = d_l^{(3)} / \zeta$$

简单地变更计算次序和加减号就可以给出逆向小波变换的提升实现为

$$s_l^{(3)} = s_l / \zeta, d_l^{(3)} = \zeta d_l$$

$$d_l^{(2)} = d_l^{(3)} - \delta s_{l+1}^{(2)}$$

$$s_l^{(1)} = s_l^{(2)} - \lambda d_l^{(2)}$$

$$d_l^{(1)} = d_l^{(2)} - \omega s_{l+1}^{(1)}$$

$$s_l^{(0)} = s_l^{(1)} - \nu d_l^{(1)}$$

$$d_l^{(0)} = d_l^{(1)} - \tau s_{l+1}^{(0)}$$

$$x_{2l} = s_l^{(0)}, x_{2l+1} = d_l^{(0)}$$

然后 根据图 14-3 和图 14-4 就可以给出相应的基于提升的正向及逆向小波变换的流程图。

以下各例都可通过类似的操作得到相应的流程图。

【例 14-7】用传统提升小波算法实现式(14-7)小波变换的提升。

$$\tilde{h}_e = h_4(z^2 + z^{-2}) + h_2(z^1 + z^{-1}) + h_0, \tilde{h}_o(z) = h_3(z^2 + z^{-1}) + h_1(z + 1)$$

这里选用小波与对偶小波都具有 4 阶消失矩的小波滤波器。

可以求出,  $\tilde{P}(z)$  存在以下因子分解:

$$\tilde{P}(z) = \begin{pmatrix} 1 & \alpha(1+z^{-1}) \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \beta(1+z) & 1 \end{pmatrix} \begin{pmatrix} 1 & \gamma(1+z^{-1}) \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \delta(1+z) & 1 \end{pmatrix} \begin{pmatrix} \zeta & 0 \\ 0 & 1/\zeta \end{pmatrix}$$

令  $r_0 = \tilde{h}_0 - 2\tilde{h}_4\tilde{h}_1/\tilde{h}_3, r_1 = \tilde{h}_2 - \tilde{h}_4 - \tilde{h}_4\tilde{h}_1/\tilde{h}_3, s_0 = h_1 - h_3 - h_3r_0/r_1, t_0 = r_0 - 2r_1$ , 则



$$\begin{aligned}\alpha &= h_4 / h_3 \approx -1.586134342 \\ \beta &= h_3 / r_1 \approx -0.05298011854 \\ \gamma &= r_1 / s_0 \approx 0.8829110762 \\ \delta &= s_0 / t_0 \approx 0.4435068522 \\ \xi &= t_0 = r_0 - 2r_1 \approx 1.149604398\end{aligned}$$

于是可以得到正向小波变换的提升实现算法为

$$\begin{aligned}s_l^{(0)} &= x_{2l}, d_l^{(0)} = x_{2l+1} \\ d_l^{(1)} &= d_l^{(0)} + \alpha(s_l^{(0)} + s_{l+1}^{(0)}) \\ s_l^{(1)} &= s_l^{(0)} + \beta(d_l^{(1)} + d_{l-1}^{(1)}) \\ d_l^{(2)} &= d_l^{(1)} + \gamma(s_l^{(1)} + s_{l+1}^{(1)}) \\ s_l^{(2)} &= s_l^{(1)} + \delta(d_l^{(2)} + d_{l-1}^{(2)}) \\ s_l &= \zeta s_l^{(2)}, d_l = d_l^{(2)} / \zeta\end{aligned}$$

【例 14-8】用提升小波算法实现小波变换的提升。

可以求出,  $\tilde{P}(z)$  的因子分解为

$$\begin{aligned}\tilde{P}(z) &= \begin{pmatrix} 1 & \alpha z^{-1} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \beta & 1 \end{pmatrix} \begin{pmatrix} 1 & \gamma z^{-1} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \delta & 1 \end{pmatrix} \\ &\times \begin{pmatrix} 1 & \varepsilon z \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \eta & 1 \end{pmatrix} \begin{pmatrix} 1 & \sigma z \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \tau & 1 \end{pmatrix} \begin{pmatrix} 1 & \lambda \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \zeta & 0 \\ 0 & 1/\zeta \end{pmatrix} \quad (14-11)\end{aligned}$$

其中,

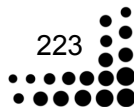
$$\begin{aligned}\alpha &= -1.58613434205942 \\ \beta &= -0.04887313389666 \\ \gamma &= 1.08844830161963 \\ \delta &= -0.58159048667007 \\ \varepsilon &= 0.12705375261988 \\ \eta &= -1.14982513039896 \\ \sigma &= -0.11439353108348 \\ \tau &= 2.15904125776103 \\ \lambda &= -0.46314 \\ \zeta &= 1.04842389233012\end{aligned}$$

由  $\tilde{P}(z)$  可以得到

$$\begin{aligned}\tilde{P}(z^{-1})^T &= \begin{pmatrix} \zeta & 0 \\ 0 & 1/\zeta \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \lambda & 1 \end{pmatrix} \begin{pmatrix} 1 & \tau \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \sigma z^{-1} & 1 \end{pmatrix} \begin{pmatrix} 1 & \eta \\ 0 & 1 \end{pmatrix} \\ &\times \begin{pmatrix} 1 & 0 \\ \varepsilon z^{-1} & 1 \end{pmatrix} \begin{pmatrix} 1 & \delta \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \gamma z & 1 \end{pmatrix} \begin{pmatrix} 1 & \beta \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \alpha z & 1 \end{pmatrix}\end{aligned}$$

于是我们可以得到正向小波变换的提升实现算法为

$$\begin{aligned}s_l^{(0)} &= x_{2l}, d_l^{(0)} = x_{2l+1} \\ d_l^{(1)} &= d_l^{(0)} + \alpha s_{l+1}^{(0)}\end{aligned}$$







$$s_l^{(1)} = s_l^{(0)} + \beta d_l^{(1)}$$

$$d_l^{(2)} = d_l^{(1)} + \gamma s_{l+1}^{(1)}$$

$$s_l^{(2)} = s_l^{(1)} + \delta d_l^{(2)}$$

$$d_l^{(3)} = d_l^{(2)} + \varepsilon s_{l-1}^{(2)}$$

$$s_l^{(3)} = s_l^{(2)} + \eta d_l^{(3)}$$

$$d_l^{(4)} = d_l^{(3)} + \sigma s_{l-1}^{(3)}$$

$$s_l^{(4)} = s_l^{(3)} + \tau d_l^{(4)}$$

$$d_l^{(5)} = d_l^{(4)} + \lambda s_l^{(4)}$$

$$s_l = \zeta s_l^{(5)}, d_l = d_l^{(5)} / \zeta$$

逆向小波变换的提升实现仿照例 14-6 中的形式，可以很容易给出，这里不再给出。

【例 14-9】用传统提升小波算法实现 D4 小波变换的提升。

D4 小波滤波器的  $z$  变换为

$$h(z) = h_0 + h_1 z^{-1} + h_2 z^{-2} + h_3 z^{-3}, g(z) = -h_3 z^2 + h_2 z^1 - h_1 + h_0 z^{-1}$$

$$\text{其中, } h_0 = \frac{1+\sqrt{3}}{4\sqrt{2}}, h_1 = \frac{3+\sqrt{3}}{4\sqrt{2}}, h_2 = \frac{3-\sqrt{3}}{4\sqrt{2}}, h_3 = \frac{1-\sqrt{3}}{4\sqrt{2}}。$$

可以求出 D4 小波滤波器的多相位矩阵在传统的提升小波算法下有如下一种因子分解：

$$P(z) = \begin{pmatrix} h_0 + h_2 z^{-1} & -h_3 z^1 - h_1 \\ h_1 + h_3 z^{-1} & h_2 z^1 + h_0 \end{pmatrix} = \begin{pmatrix} 1 & -\sqrt{3} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \frac{\sqrt{3}}{4} + \frac{\sqrt{3}-2}{4} z^{-1} & 1 \end{pmatrix} \begin{pmatrix} 1 & z \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{\sqrt{3}+1}{\sqrt{2}} & 0 \\ 0 & \frac{\sqrt{3}-1}{\sqrt{2}} \end{pmatrix}$$

由此，可以给出 D4 小波变换的传统提升实现算法为

$$s_l^0 = x_{2l}, d_l^0 = x_{2l+1}$$

$$s_l^1 = s_l^0 + \sqrt{3} d_l^0$$

$$d_l^1 = d_l^0 - \frac{\sqrt{3}}{4} s_l^1 - \frac{\sqrt{3}-2}{4} s_{l-1}^1$$

$$s_l^2 = s_l^1 - d_{l+1}^1$$

$$s_l = \frac{\sqrt{2}}{\sqrt{3}+1} s_l^2, d_l = \frac{\sqrt{2}}{\sqrt{3}-1} d_l^2$$

【例 14-10】用提升小波算法实现 D4 小波变换的提升。

用改进的 Laurent 多项式 Euclidean 算法计算出

$$P(z) = \begin{pmatrix} h_0 + h_2 z^{-1} & -h_3 z^1 - h_1 \\ h_1 + h_3 z^{-1} & h_2 z^1 + h_0 \end{pmatrix}$$



$$= \begin{pmatrix} 1 & -\sqrt{3} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \frac{\sqrt{3}-2}{4}z^{-1} & 1 \end{pmatrix} \begin{pmatrix} 1 & \frac{2}{\sqrt{3}} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \frac{\sqrt{3}}{2} & 1 \end{pmatrix} \begin{pmatrix} 1 & \frac{1}{4}z - \frac{1}{\sqrt{3}} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{1+\sqrt{3}}{2\sqrt{2}} & 0 \\ 0 & \frac{2\sqrt{2}}{1+\sqrt{3}} \end{pmatrix}$$

由此，可以给出 D4 小波变换的本文提升实现算法为

$$\begin{aligned} s_l^0 &= x_{2l}, d_l^0 = x_{2l+1} \\ s_l^1 &= s_l^0 + \sqrt{3}d_l^0 \\ d_l^1 &= d_l^0 - \frac{\sqrt{3}-2}{4}s_{l-1}^1 \\ s_l^2 &= s_l^1 - \frac{2}{\sqrt{3}}d_l^1 \\ d_l^2 &= d_l^1 - \frac{\sqrt{3}}{2}s_l^2 \\ s_l^3 &= s_l^2 - \frac{1}{4}d_{l+1}^2 + \frac{1}{\sqrt{3}}d_l^2 \\ s_l &= \frac{\sqrt{3}+1}{2\sqrt{2}}s_l^3, d_l = \frac{2\sqrt{2}}{\sqrt{3}+1}d_l^3 \end{aligned}$$

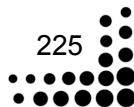
从上面的例子可以看到，对于具有线性相位的小波滤波器，它们的  $K^2s(z)$  为常数；而对于对称性不好的 D4 小波滤波器而言， $K^2s(z)$  为多项式。

还可以看出，除了最后一步提升外，其他的提升和对偶提升步骤都是用奇序列中的一项来预测偶序列中的一项，或用偶序列中的一项来更新奇序列中的一项。计算过程只是用简单的乘法运算和加减运算得到，计算复杂度非常低。对于具有线性相位的小波滤波器，所有的提升步骤都是通过简单单项乘法及加法得到，完全去除了传统提升运算中的卷积运算，更加有利于软硬件的实现。

另外，当具有线性相位的滤波器用于处理信号时，采用对称周期延拓的信号边界处理方法不仅可实现小波变换的完全重构，而且不增加变换后的数据量。

## 14.8 整数小波变换

数字图像都是用整数来表示其像素值的，而小波滤波器表示却具有浮点数系数。对数字图像进行小波变换处理后数据，即得到的小波系数不再为整数。由于用二进制来表示浮点数的精度有限，因而经过小波变换后得到的小波系数重构时，就会有能量损失。因此，当人们想对数字图像进行无损压缩时，就希望变换后的小波系数仍为整数。提升算法可以十分方便地构造整数到整数的小波变换。将整数小波变换应用于图像压缩就可以实现无失真的图像压缩。该算法是通过在忽略归一化因子的情况下，将算子  $\lfloor x+1/2 \rfloor$  作用于每个提升步骤中的算子  $u_i(z)$  和  $p_i(z)$  实现的。而向下取整算子“ $\lfloor \rfloor$ ”是非线性的，因此，整数小波变换是非线性变换。利用例 14-8 中的结果，给出小波变换基于提升小波算法的整数提升形式为





$$\begin{aligned}
s_l^{(0)} &= x_{2l}, d_l^{(0)} = x_{2l+1} \\
d_l^{(1)} &= d_l^{(0)} + \lfloor \alpha s_{l+1}^{(0)} + 1/2 \rfloor \\
s_l^{(1)} &= s_l^{(0)} + \lfloor \beta d_l^{(1)} + 1/2 \rfloor \\
d_l^{(2)} &= d_l^{(1)} + \lfloor \gamma s_{l+1}^{(1)} + 1/2 \rfloor \\
s_l^{(2)} &= s_l^{(1)} + \lfloor \delta d_l^{(2)} + 1/2 \rfloor \\
d_l^{(3)} &= d_l^{(2)} + \lfloor \varepsilon s_{l+1}^{(2)} + 1/2 \rfloor \\
s_l^{(3)} &= s_l^{(2)} + \lfloor \eta d_l^{(3)} + 1/2 \rfloor \\
d_l^{(4)} &= d_l^{(3)} + \lfloor \sigma s_{l+1}^{(3)} + 1/2 \rfloor \\
s_l^{(4)} &= s_l^{(3)} + \lfloor \tau d_l^{(4)} + 1/2 \rfloor \\
d_l^{(5)} &= d_l^{(4)} + \lfloor \lambda s_l^{(4)} + 1/2 \rfloor \\
s_l &= \zeta s_l^{(5)}, d_l = d_l^{(5)} / \zeta
\end{aligned}$$

通过上面的讨论我们可以知道，从多分辨分析的尺度函数或直接从滤波器  $h$  出发可以构造出具有紧支撑性的双正交小波基，并在传统小波变换的提升算法的基础上，给出了一种新的提升算法。该算法在计算复杂度上比传统算法略微下降，但在软硬件实现方面更加简单。提升的实现形式给出了小波完全的空间域解释，它具有许多优良的特性：结构简单、运算量低、原位运算、节省存储空间、逆变化可以直接反转实现以及可逆的整数到整数变换，便于实现，在高速处理、移动手持设备、低功耗设备应用中具有很大的吸引力。提升小波在 1996 年由 Sweldens 提出后，在信号处理领域得到了广泛的应用。在静态图像处理中，提升小波已被选作 JPEG 2000 的变换核。它提供了多精度的功能，同基于 JPEG 2000 的标准相比，在很低的比特率时具有较好的压缩 DCT 的 JPEG 性能，并且提供了在同一个编码结构内有效的失真和无失真压缩。在视频领域，使用提升小波方法自适应地对任意形状的物体进行编码，显著提高了编码效率，在静止图像编码上明显优于 MPEG 4；视频物体的主观评价效果更好，具有比 MPEG 4 更少的块效应。通过提升小波的梯形结构，提出渐进性的小波逆变换合成 (PIWC) 算法来保证一个局域场景的再现只需要使用部分压缩数据，这就减少了数据访问量 and 计算开销，实现了在 3D 环境下从压缩数据中实时再现 3D。提升小波用于一维信号去噪和图像去噪也取得了良好的效果，通过将水印加入提升结构正在处理的小波系数中，进一步增强了安全性。

# 第15章 基于小波的阈值去噪 方法分析

阈值去噪方法就是在小波分解后的各层系数中，对模大于或小于某阈值  $T$  的系数分别处理，然后对处理完的小波系数再反变换重构出经去噪后的信号。在阈值去噪中，阈值函数体现了对超过和低于阈值的小波系数模的不同处理策略以及不同的估计方法。

小波分析是近年来迅速发展起来的新兴学科，已经成为众多学科共同关注的热点。目前小波分析正逐步应用于信号分析、系统控制、图像处理、量子力学、计算识别、语音识别和合成以及故障诊断等领域。本章研究小波分析在信号阈值去噪方面的应用。

## 15.1 阈值去噪方法

设信号  $f$  被加性噪声  $W$  污染，则观测信号表示为

$$X = f + W \quad (15-1)$$

从统计学的观点看，去噪是用实际上可观测到的数据  $X$  对信号  $f$  的估计。在正交规范基  $B = \{g_m\}_{0 \leq m \leq N-1}$  下观测信号  $X$  被分解为

$$X_B(m) = f_B(m) + W_B(m)$$

显然有

$$X_B(m) = \langle X, g_m \rangle, f_B(m) = \langle f, g_m \rangle, W_B(m) = \langle W, g_m \rangle$$

假设  $W$  是方差为  $\sigma^2$  的零均匀白噪声，即

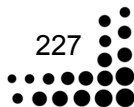
$$E[W(n)W^*(k)] = \sigma^2 \delta(n-k) \quad (15-2)$$

则  $E[W_B(m)] = 0$ ，且有

$$\begin{aligned} E[W_B(m)W_B^*(p)] &= \sum_{n=0}^{N-1} \sum_{k=0}^{N-1} g_m^*(n)g_p(k)E[W(n)W^*(k)] \\ &= \sigma^2 \sum_{n=0}^{N-1} g_m^*(n)g_p(n) = \sigma^2 \delta(p-m) \end{aligned}$$

也就是说，方差为  $\sigma^2$  的零均值白噪声在变换域的内积系数仍然是方差为  $\sigma^2$  的零均值白噪声，所以白噪声的能量在变换域是均匀分布的。

由于在变换域内信号能量集中在少量大幅值内积系数上，而噪声能量在变换域却是均匀分布的，所以可以从一个特定的  $X_B(m)$  来独立地估计  $f_B(m)$ ，从而得到信号的估计





$$\tilde{F} = DX = \sum_{m=0}^{N-1} d_m [X_B(m)] g_m \quad (15-3)$$

式中,  $D$  称为对角估计算子。因为噪声均值为零, 所以当观测信号为零时, 估计信号为零显然是合理的, 故  $D0=0$ , 因而  $d_m(0)=0$ , 于是对角估计函数可以表示为

$$d_m(X_B(m)) = \alpha(m)X_B(m), \quad 0 \leq m \leq N-1$$

估计的均方误差称为风险, 它可以用内积系数表示为

$$r(D, f) = E \left\{ \|f - \tilde{F}\|^2 \right\} = \sum_{m=0}^{N-1} E \left\{ |f_B(m) - \alpha(m)X_B(m)|^2 \right\} \quad (15-4)$$

线性估计假定信号能量集中在前  $M$  个数上, 于是当  $0 \leq m \leq M-1$  时, 选择  $\alpha(m)=1$ ; 而当  $M \leq m \leq N-1$  时,  $\alpha(m)=0$ 。由 (15-4) 式得线性估计的风险为

$$r(D, f) = M\sigma^2 + \sum_{m=M}^{N-1} |f_B(m)|^2 = M\sigma^2 + \varepsilon_l(M)$$

上式第 1 项为噪声带来的估计误差, 第 2 项为线性逼近误差。 $M$  越大, 线性逼近误差越小, 但噪声带来的估计误差越大; 反之,  $M$  越小, 噪声带来的估计误差越小, 但线性逼近误差越大。所以应调整  $M$  使这两项具有相同数量级。显然, 线性估计不是最优的, 尤其对非正则信号更是如此。

阈值估计有硬阈值估计和软阈值估计两种。硬阈值函数为

$$d_m(x) = h_T(x) = \begin{cases} x, & |x| > T \\ 0, & |x| \leq T \end{cases}$$

而软阈值函数为

$$d_m(x) = s_T(x) = \begin{cases} x - T, & x > T \\ x + T, & x < -T \\ 0, & |x| \leq T \end{cases}$$

显然, 硬阈值函数有间断点, 它只是简单地保留或者去掉信号。而软阈值在进行阈值判别的同时也用阈值  $T$  对信号进行衰减。

## 15.2 阈值风险

上节讨论硬阈值估计和软阈值估计时尚未涉及阈值如何确定的问题, Donoho 和 Johnstone 已证明: 对适当选取的阈值, 阈值估计的风险接近  $r_p(f)$ 。用  $O_d$  表示所有对角算子的集合,  $N$  表示信号长度, 关于阈值估计风险有下述定理。

定理 1 设  $T = \sigma\sqrt{2\log_2 N}$ , 对所有  $N \geq 4$ , 硬或软阈值估计风险  $r_t(f)$  满足

$$r_t(f) \leq (2\log_2 N + 1) [\sigma^2 + r_p(f)] \quad (15-5)$$

在所有  $B$  的对角估计算子中, 因子  $2\log_2 N$  是最优的, 即

$$\lim_{N \rightarrow +\infty} \inf_{D \in O_d} \sup_{f \in C^N} \frac{E \left\{ \|f - DX\|^2 \right\}}{\sigma^2 + r_p(f)} \frac{1}{2\log_2 N} = 1$$



该定理说明, 阈值估计风险最多比  $r_p(f)$  大  $2\log_2 N$ , 且因子  $2\log_2 N$  不能由任何其他对角估计算子所改进。

还可以证明, 用  $r_{\inf}(f)$  代替  $r_p(f)$  时, 上述定理仍然成立。

按上述定理选取的阈值  $T > \sigma$ , 即平均而言, 阈值高于噪声电平, 或者说阈值以极大的概率高于噪声电平, 所以阈值与  $\sigma$  成正比是不难理解的。但阈值为什么会随样本数  $N$  的增大而增大呢? 这是由于高斯分布的截断会随  $N$  的增加而产生越来越大的噪声系数。为什么  $r_t(f)$  的上界会出现因子  $2\log_2 N$  呢? 当  $|f_B(m)| < \sigma$  时, 由于  $T > \sigma$ ,  $|X_B(m)| < T$  的可能性很大, 硬阈值估计和 Oracle 投影阈值估计都将  $X_B(m)$  置零, 不会造成  $r_t(f)$  与  $r_p(f)$  的很大差别; 当  $|f_B(m)| > 2T$  时,  $|X_B(m)| > T$  的可能性很大, 硬阈值估计和 Oracle 投影阈值估计都将保留  $X_B(m)$ , 也不会造成  $r_t(f)$  与  $r_p(f)$  的很大差别; 而当  $\sigma < |f_B(m)| < 2T$  时, Oracle 投影阈值估计产生的风险为  $\min[|f_B(m)|^2, \sigma^2] = \sigma^2$ , 若此时  $|X_B(m)| > T$ , 则硬阈值估计将  $X_B(m)$  置零, 产生的风险为

$$|f_B(m)|^2 \approx T^2 = 2\sigma \log_2 N$$

也就是说, 硬阈值估计的风险比 Oracle 投影阈值估计大  $2\log_2 N$  倍。一般来说,  $|f_B(m)|$  与  $T$  为同一数量级的部分还是很小的, 所以硬阈值估计风险  $r_t(f)$  比式 (15-5) 所示的上界小得多。在这个意义下, 硬阈值估计风险比  $r_t(f)$  接近理论极限  $r_{\inf}(f)$ , 是一种最优估计算法。

**定理 2** 假定噪声是白噪声, 如噪声是有色噪声, 则  $E\{|W_B(m)|^2\} = \sigma_m^2$ 。这时仍可用阈值估计算法去噪, 且有如下定理:

**定理 3** 对  $0 \leq m \leq N-1$ , 取  $T_m = \sigma_m \sqrt{2\log_2 N}$ , 对所有  $N \geq 4$ , 硬或软阈值估计风险  $r_t(f)$  满足

$$r_t(f) \leq (2\log_2 N + 1) [\bar{\sigma}^2 + r_p(f)]$$

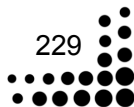
其中,  $\bar{\sigma}^2 = \frac{1}{N} \sum_{m=0}^{N-1} \sigma_m^2$ 。

在实际中,  $T$  的估计应是自适应的, 即应考虑到信号的相对平稳性和信噪比的大小。对于平稳性较差的信号,  $T$  值应选得小一些, 相反情况  $T$  值应取得较大。对同一信号, 信噪比大时, 噪声功率小, 则  $T$  值取得小一些。

## 15.3 实验结果与分析

一维信号利用小波进行除噪的步骤如下。

- (1) 对含噪信号进行预处理, 便于后续处理。
- (2) 一维信号的小波分解。选择一个小波并确定一个分解层次  $N$ , 然后对信号  $s$  进行  $N$  层分解。
- (3) 小波分解高频系数的阈值量化。对第  $1 \sim N$  层的每层高频系数都选择一个软阈值或硬阈值量化处理。
- (4) 一维小波的重构。根据小波分解的第  $N$  层的系数和经过量化处理后的第  $1 \sim N$  层的





每层高频系数，都进行信号的重构。

从上面的步骤可以看出，最关键的是如何选择阈值进行阈值量化，在某种程度上，它直接关系到信号降噪的质量。

在实际仿真程序中，小波阈值处理一般有下列 3 种方法。

(1) 默认阈值去噪处理。该方法利用一个函数生成默认阈值，然后利用一个函数进行去噪处理。

(2) 给定阈值去噪处理。在实际的去噪处理过程中，阈值往往可通过经验公式获得，且这种阈值比默认阈值的可信度高。

(3) 强制去噪处理。该方法是将小波分解结构中的高频系数全部变为零，然后对信号进行重构处理。这种方法比较简单，且去噪后的信号比较平滑，但是容易丢失信号中的有用成分。

### 15.3.1 利用小波分析对含噪正弦波进行去噪

【例 15-1】利用小波分析对含噪正弦波进行去噪。

```
%生成正弦信号
N=1000;
t=1:N;
x=sin(0.03*t);
%加噪声
load noissin;
ns=noissin;
%显示波形
subplot(3,1,1);
plot(t,x);
title('(a)原始信号');
ylabel('幅值 A');
subplot(3,1,2);
plot(ns);
title('(b)含噪信号');
ylabel('幅值 A');
%小波去噪
xd=wden(ns,'minimaxi','s','one',5,'db3');
subplot(3,1,3);
plot(xd);
title('(c)去噪信号');
ylabel('幅值 A');
```

程序运行结果如图 15-1 所示。

在这个仿真程序中，展示了从噪声信号中恢复信号的问题，其中，图 15-1 (b) 所示是含噪信号。从图 15-1 (c) 中可以看出，去噪后的信号大体上恢复了原始信号的形状，并明显地除去了噪声所引起的干扰。但是恢复后的信号（见图 15-1 (c)）和原始信号（见图 15-1 (a)）



相比，有明显的改变。这主要是在进行去噪处理的过程中所用的分析小波和细节系数阈值不恰当所致。

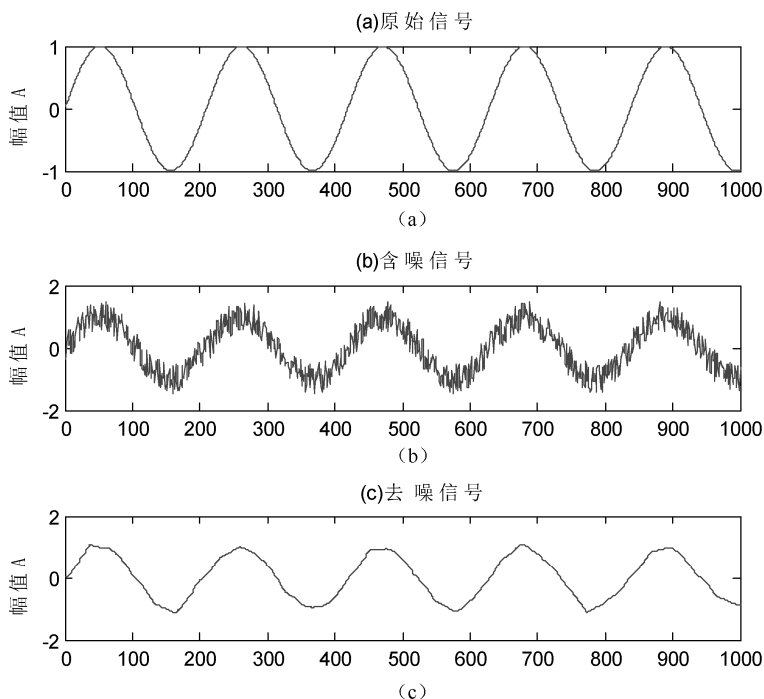


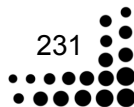
图 15-1 原信号、含噪信号与去噪信号

### 15.3.2 小波分析对污染信号进行去噪处理

在电网电压值监测过程中，有时会由于监测设备出现了一点故障，致使所采集到的信号受到噪声的污染。现在利用小波分析对污染信号进行去噪处理以恢复原始信号。

**【例 15-2】**小波分析对污染信号进行去噪处理。

```
%装载采集的信号 leleccum.mat
load leleccum;
%将信号中第 2000 ~ 3450 个采样点赋给 s
indx=2000:3450;
s=leleccum(indx);
%画出原始信号
subplot(2,2,1);
plot(s);
title('(a)原始信号');
ylabel('幅值 A');
%用 db1 小波对原始信号进行 3 层分解并提取系数
[c,l]=wavedec(s,3,'db1');
```







```
a3=appcoef(c,l,'db1',3);
d3=detcoef(c,l,3);
d2=detcoef(c,l,2);
d1=detcoef(c,l,1);
%对信号进行强制性去噪处理并图示结果
dd3=zeros(1,length(d3));
dd2=zeros(1,length(d2));
dd1=zeros(1,length(d1));
c1=[a3 dd3 dd2 dd1];
s1=waverec(c1,l,'db1');
subplot(2,2,2);
plot(s1);
title('(b)强制去噪后的信号');
ylabel('幅值 A');
%用默认阈值对信号进行去噪处理并图示结果
%用 ddencmp 函数获得信号的默认阈值
[thr,sorh,keepapp]=ddencmp('den','wv',s);
s2=wdencmp('gbl',c,l,'db1',3,thr,sorh,keepapp);
subplot(2,2,3);
plot(s2);
title('(c)默认阈值去噪后的信号');
ylabel('幅值 A');
%用给定的软阈值进行去噪处理
softd1=wthresh(d1,'s',1.465);
softd2=wthresh(d2,'s',1.823);
softd3=wthresh(d3,'s',2.768);
c2=[a3 softd3 softd2 softd1];
s3=waverec(c2,l,'db1');
subplot(2,2,4);
plot(s3);
title('(d)给定软阈值去噪后的信号');
ylabel('幅值 A');
```

程序运行结果如图 15-2 所示。

在本例中，分别利用强制阈值去噪[见图 15-2 (b)]、默认阈值去噪[见图 15-2 (c)]和给定软阈值去噪[见图 15-2 (d)]方法进行处理。从图 15-2 得到的结果来看，应用强制去噪处理后的信号较光滑，但它很有可能丢失了信号中的一些有用的成分。默认阈值去噪和给定软阈值去噪两种处理方法在实际中应用得更为广泛一些。

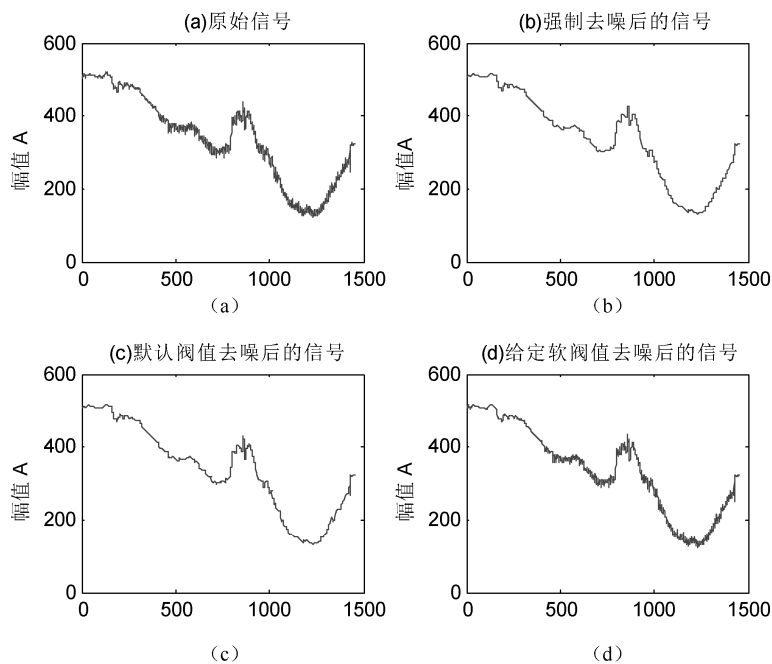
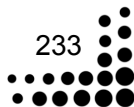


图 15-2 3 种不同去噪方法的结果

### 15.3.3 利用软、硬阈值去噪

【例 15-3】利用软、硬阈值去噪。

```
%生成线性信号
y=linspace(-1,1,100);
%设置 T 阈值
thr=0.4;
%计算软、硬阈值
ythard=wthresh(y,'h',thr);
ytsoft=wthresh(y,'s',thr);
%显示不同阈值后的信号
subplot(1,3,1);
plot(y);
title('原始信号');
xlabel('样本序号 n');
ylabel('幅值 A');
subplot(1,3,2);
plot(ythard);
title('硬阈值信号');
xlabel('样本序号 n');
ylabel('幅值 A');
subplot(1,3,3);
```





```
plot(ytsoft);  
title('软阈值信号');  
xlabel('样本序号 n');  
ylabel('幅值 A');
```

程序运行结果如图 15-3 所示。

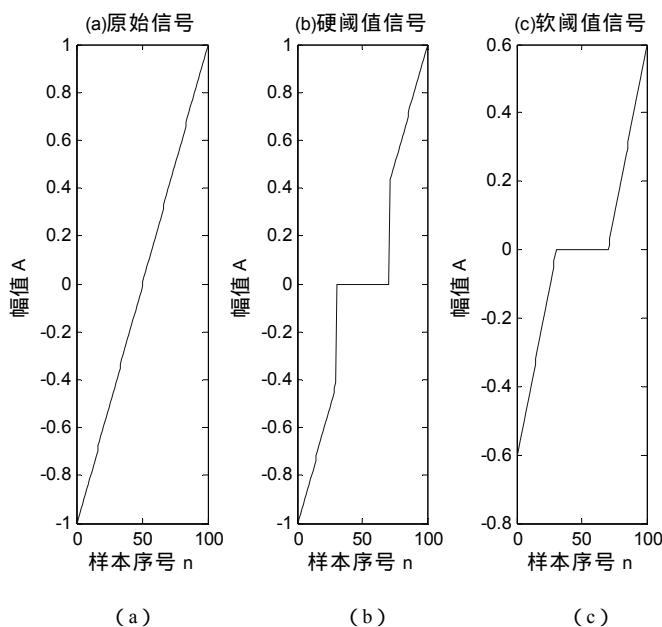


图 15-3 不同阈值下的信号

图 15-3 (b) 和图 15-3 (c) 所示分别是原始信号 (见图 15-3 (a)) 经过软、硬阈值变换后的图形。图 15-3 (b) 中, 把信号的绝对值与指定的阈值进行比较, 小于或等于阈值的点变为零, 大于阈值的点保持不变。图 15-3 (c) 中, 把信号的绝对值与指定的阈值进行比较, 小于或等于阈值的点变为 0, 大于阈值的点变为该点值与阈值的差。一般来说, 硬阈值比软阈值处理后的信号要粗糙一些。从图 15-3 中可以看出, 最关键的是如何选择阈值进行阈值量化, 它直接关系到信号降噪的质量。

在实际的工程应用中, 所分析的信号可能包含许多尖峰或突变部分, 并且噪声也不是平稳的白噪声, 对这种信号进行分析, 首先需要做信号的预处理, 将信号噪声部分去除, 提取有用信号。对于这种信号的去噪, 用传统的傅里叶变换分析显得无能为力, 因为傅里叶分析是将信号完全在频率域中进行分析, 不能给出信号在某个时间点的变化情况, 使得信号在时间轴上的任何一个突变都会影响信号的整个频谱。小波分析由于能同时在时频域中对信号进行分析, 具有多分辨率分析功能, 所以能在不同的分解层上有效地区分信号的突变部分和噪声, 从而实现信号的去噪。

# 第16章 连续与离散小波算法 分析与实现

## 16.1 信号分解

### 16.1.1 信号的连续小波分解

本小节首先介绍 MATLAB 中的连续小波分解函数,然后讲述在 MATLAB 命令行工具中,如何利用这些函数实现一维连续分解变换。

#### 1. 一维连续小波分解函数

在 MATLAB 中实现一维小波分解的函数是 `cwt`,其调用格式有以下 4 种:

```
COEFS=cwt(S,SCALES,'wname')
COEFS=cwt(S,SCALES,'wname','plot')
COEFS=cwt(S,SCALES,'wname','PLOTMODE')
COEFS=cwt(S,SCALES,'wname','PLOTMODE',XLIM)
```

**格式** 采用'wname'小波,在正、实尺度 SCALES 下计算向量一维小波系数。 $S$  为被分析信号,小波可以是实数,也可以是虚数。

**格式** 除了计算小波系数外,还加以图形显示。

**格式** 计算并画出连续小波变换的系数,并使用 PLOTMODE 对图形着色。表 16-1 所示是 PLOTMODE 字符串的有效值。

**格式** 相当于格式 中的语法 `COEFS=cwt(S,SCALES,'wname','absglb')`。

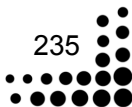
在 PLOTMODE 参数列前面加'3D',使用同样的关键字可以得到 3-D 图形,如 `COEFS=cwt(S,SCALES,'wname','3Dplot')`或 `COEFS=cwt(S,SCALES,'wname','3Dlvt')`。

**格式** 能够计算并画出连续小波变换系数。系数使用 PLOTMODE 和 XLIM 进行着色。其中,  $XLIM=[x1,x2]$ ,并且  $1 \leq x1 \leq x2 \leq \text{length}(S)$ 。

假设  $s$  是信号,  $\psi$  是小波,则对应于尺度  $a$  和位置  $b$  的小波系数为

$$C_{a,b} = \int_R s(t) \frac{1}{\sqrt{a}} \overline{\psi\left(\frac{t-b}{a}\right)} dt$$

由于  $s(t)$  是离散信号,我们用  $s(k)$  的形式表示,  $k=0,1,\cdots,\text{length}(s)$ 。对于每个尺度  $a$ ,





都从  $b = 1 \sim ls = \text{length}(s)$  计算相应的小波系数  $C_{a,b}$ ，并存储到  $\text{COEFS}(i,:)$  中， $a = \text{SCALES}(i)$ 。

表 16-1 PLOTMODE 字符串的有效值

MODE 值	含 义
'lvl'	逐一尺度着色模式
'glb'	全尺度着色模式
'abslvl'或'lvlabs'	按系数绝对值逐一着色模式
'absglb'或'glbabs'	按系数绝对值全尺度着色模式

输出变量 COEFS 是一个  $la \times ls$  矩阵。这里， $la$  是 SCALES 的长度；COEFS 是一个实矩阵或者复矩阵，主要和小波类型有关。

【例 16-1】cwt 函数应用举例。

```
t=linspace(-1,1,512);  
s=1-abs(t); c=cwt(s,1:32,'cgau4');  
c=cwt(s,[64 32 16:-2:2],'morl');  
c=cwt(s,[3 18 12.9 7 1.5],'db2');  
c=cwt(s,1:64,'sym4','abslvl',[100:400]);
```

程序运行结果如图 16-1 所示。

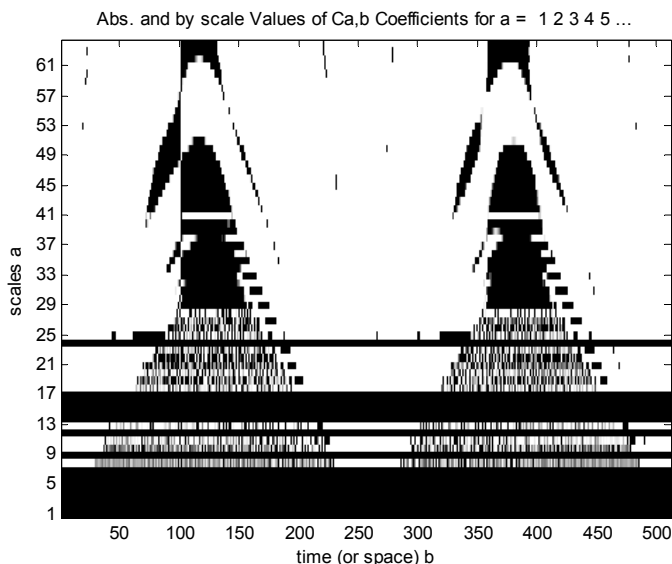


图 16-1 cwt 函数应用举例

【例 16-2】以信号 noissin 为例，说明如何对一个信号进行连续小波分解，信号 noissin 是一个含噪的周期性信号。

```
% 装载 noissin 信号  
load noissin;  
x=noissin;  
figure(1);
```

```

plot(x);
figure(2);
subplot(121);
% 用 db4 小波函数进行一维连续小波变换
c=cwt(x,1:48,'db4','plot');
subplot(122);
% 重新选择尺度后进行一维连续小波变换
c=cwt(x,2:2:128,'db4','plot');
figure(3);
% 使用复小波对其进行连续小波变换
c=cwt(x,2:2:128,'cgau4','plot');

```

程序运行结果如图 16-2 ~ 图 16-4 所示。

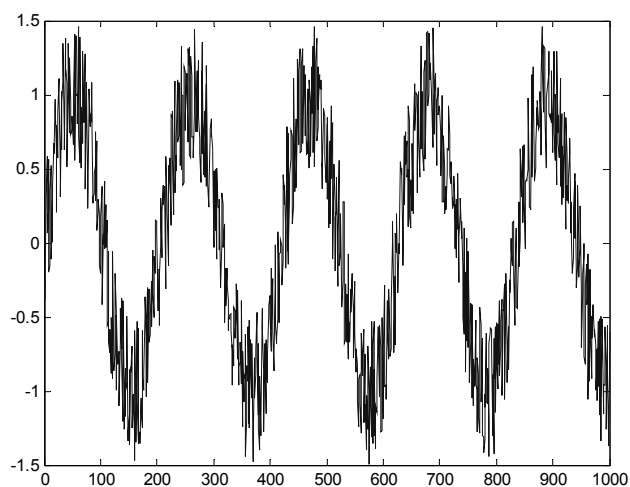


图 16-2 原始信号

Absolute Values of  $C_{a,b}$  Coefficients for  $a = 1\ 2\ 3\ 4\ 5\ \dots$  Absolute Values of  $C_{a,b}$  Coefficients for  $a = 2\ 4\ 6\ 8\ 10\ \dots$

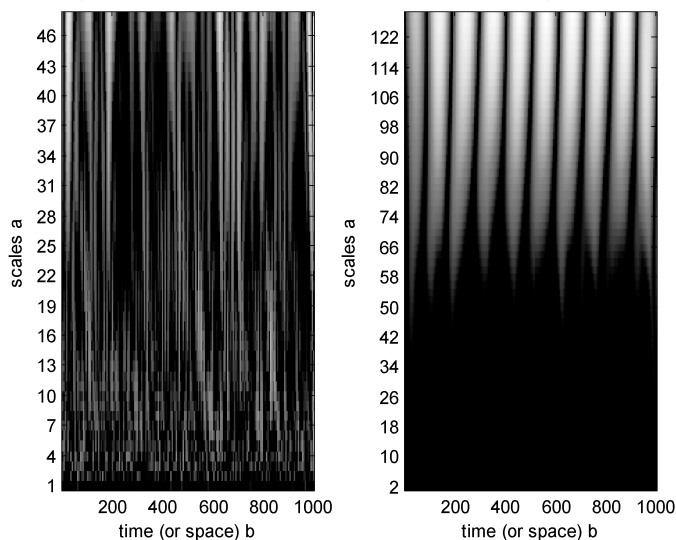


图 16-3 不同尺度下的连续小波分解

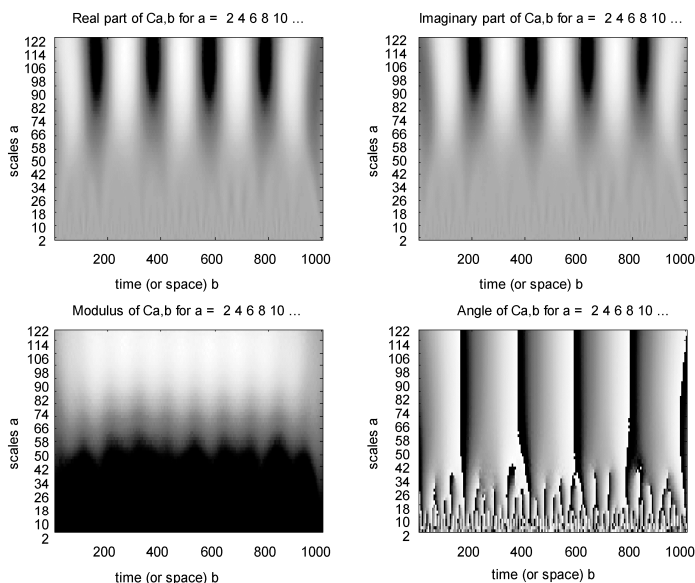


图 16-4 信号的复连续小波分解

## 2. 提取一维小波细节系数函数

在 MATLAB 中实现提取一维小波细节系数的函数是 `detcoef`，其调用格式有如下两种：

```
D=detcoef(C,L,N)
```

```
D=detcoef(C,L)
```

**格式** 由小波分解结构 `[C,L]` 提取 `N` 层细节系数。

`N` 必须是正整数，并且  $1 \leq N \leq NMAX$ ， $NMAX=length(L)-2$ 。

**格式** 提取最后一层 `NMAX` 的细节系数。如果 `N` 是由正整数组成的向量，并且  $1 \leq N(j) \leq NMAX$ ，则 `DECELL=detcoef(C,L,N,'cells')` 返回一个单元阵列，`DCELL{j}` 包含细节 `N(j)` 的系数。`DCELL=detcoef(C,L,'cells')` 等同于 `DCELL=detcoef(C,L,[1:NMAX])`。`[D1,...,Dp]=detcoef(C,L,[N(1),...,N(p)])` 提取层 `[N(1),...,N(p)]` 的细节系数。

**【例 16-3】** `detcoef` 函数应用举例。

```
% 采用补零的扩展模式
% 装载一维尺度信号
load leleccum;
s=leleccum(1:3920);
% 使用 db1 在第 3 层进行分解
[c,l]=wavedec(s,3,'db1');
subplot(411);plot(s);
title('原始信号');
% 从小波分解结构中提取 1~3 层的细节系数
[cd1,cd2,cd3]=detcoef(c,l,[1 2 3]);
% 绘图命令
subplot(412);plot(cd3);Ylabel('cd3');
subplot(413);plot(cd2);Ylabel('cd2');
```



```
subplot(414);plot(cd1);Ylabel('cd1');
```

程序运行结果如图 16-5 所示。

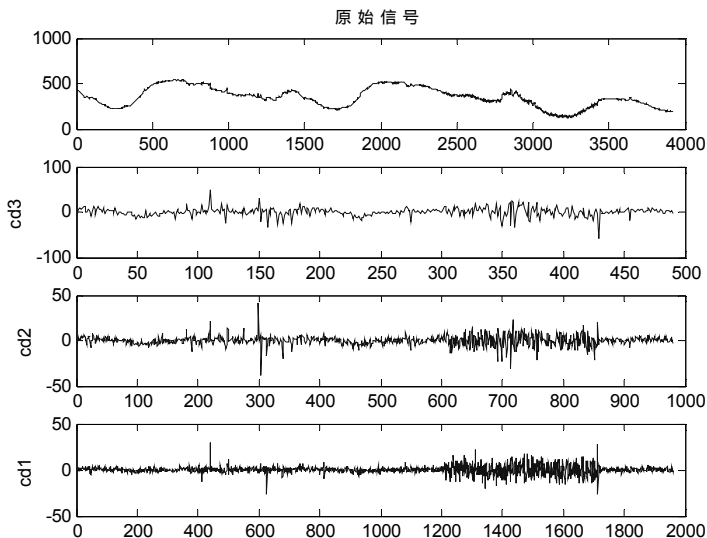


图 16-5 函数应用举例

### 3. 提取一维近似系数函数

在 MATLAB 中实现提取一维近似系数的函数是 `appcoef`，其调用格式有以下 4 种：

```
A=appcoef(C,L,'wname',N)
A=appcoef(C,L,'wname')
A=appcoef(C,L,Lo_R,Hi_R)
A=appcoef(C,L, Lo_R,Hi_R,N)
```

**格式** 使用小波分解框架 `[C,L]` 计算 `N` 层系数的近似值。'wname' 是包含小波名的字符串。层数 `N` 必须是正整数，并且  $0 < N \leq \text{length}(L)-2$ 。

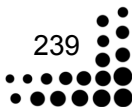
**格式** 提取最大层，即  $\text{length}(L)-2$  层的系数近似值。

除了直接给定小波名以外，还可以采用给定滤波器的方法。格式 、 中，`Lo_R` 是重构低通滤波器，`Hi_R` 是重构高通滤波器。

**算法分析：**输入变量 `C` 和 `L` 包含信号分解的所有信息。令  $N_{\text{MAX}} = \text{length}(L)-2$ ，那么  $C = [A(N_{\text{MAX}}) D(N_{\text{MAX}}) \dots D(1)]$ ，其中 `A` 和 `D` 都是向量。如果  $N = N_{\text{MAX}}$ ，那么做简单提取；否则，`appcoef` 采用反小波变换反复计算系数近似值。

**【例 16-4】** `appcoef` 函数应用举例。

```
% 采用补零的扩展模式
% 装载一维尺度信号
load leleccum;
s=leleccum(1:3920);
subplot(211);plot(s);
title('原始信号');
```







```
% 使用 db1 在第 3 层进行分解
[c,l]=wavedec(s,3,'db1');
% 由小波分解框架[c,l]提取第 3 层系数近似值
ca3=appcoef(c,l,'db1',3);
subplot(2,1,2);plot(ca3);
```

程序运行结果如图 16-6 所示。

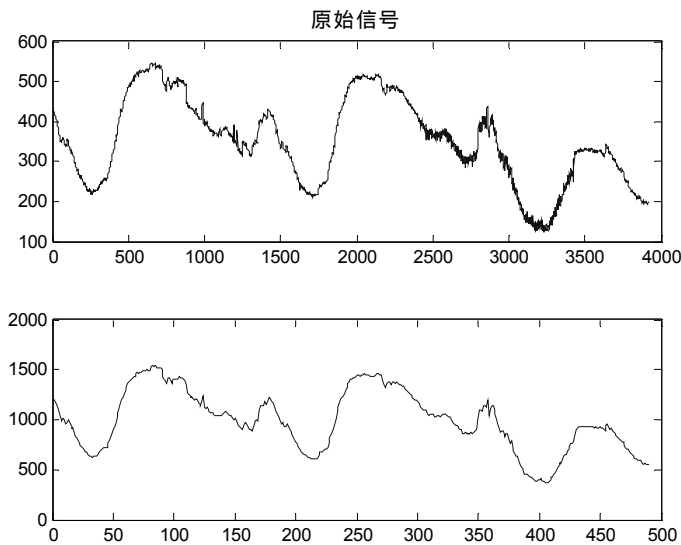


图 16-6 appcoef 函数应用举例

#### 4. 一维单尺度小波分解函数

在 MATLAB 中实现一维单尺度小波分解的函数是 upwlev，其调用格式有如下两种：

```
[NC,NL,cA]=upwlev(C,L,'wname')
[NC,NL,cA]=upwlev(C,L,Lo_R,Hi_R)
```

格式 用于对小波分解结构[C,L]进行单尺度重构，返回新的结构 [NC,NL] 并提取最后一尺度的低频系数向量 cA。

如果[C,L]是  $n=\text{length}(L)-2$  尺度的分解结构，那么 [NC,NL] 是尺度  $n-1$  的分解结构，cA 是尺度  $n$  的低频系数向量。

'wname'是包含小波名的字符串；C 是原始小波分解向量；L 是相应的记录向量。

除了给定小波名，还可以给定滤波器。对于格式 ，Lo\_R 是重构低通滤波器，Hi\_R 是重构高通滤波器。

【例 16-5】upwlev 函数应用举例。

```
% 采用补零的扩展模式
% 装载一维尺度信号
load sumsin;s=sumsin;
% 使用 db1 在第 3 层小波分解
[c,l]=wavedec(s,3,'db1');
```



```
subplot(311);plot(s);
title('原始信号');
subplot(312);plot(c);
title('3 层小波分解结构');
xlabel(['第 3 层的低频系数以及第 3 ~ 1 层的高频系数'])
% 对 3 层的小波分解结构进行重构
% 因此新的结构[nc,nl]是 2 层小波分解结构
[nc,nl]=upwlev(c,l,'db1');
subplot(313);plot(nc);
title('2 层小波分解结构')
xlabel(['第 2 层的低频系数及第 1、2 层的高频系数'])
```

程序运行结果如图 16-7 所示。

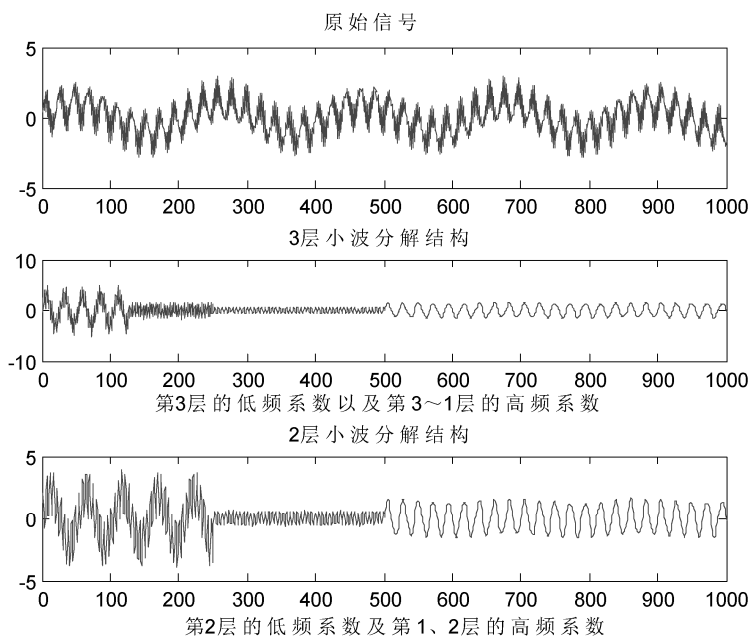


图 16-7 upwlev 函数应用举例

### 5. 小波分解的最大尺度函数

在 MATLAB 中实现小波分解的最大尺度的函数是 wmaxlev，其调用格式如下：

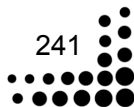
```
L=wmaxlev(S,'wname')
```

该函数是一维或者二维小波函数或小波包的导向函数，它返回信号或图像的最大分解尺度，使用它可以避免分解时超过这个值。

一般情况下，小波分解尺度都小于理论上的最大值。通过一维分解时分解尺度为 5，而二维分解尺度为 3。

【例 16-6】计算一维信号和二维信号分解的最大尺度。

```
% 对于一维信号
```





```
s=2^10;  
w='db1';  
% 计算小波分解的最大尺度  
l1=wmaxlev(s,w)  
% 改变小波  
w='db7';  
% 计算最大分解尺度  
l2=wmaxlev(s,w)  
% 对于二维信号  
s=[2^9 2^7];  
w='db1';  
% 计算最大分解尺度  
l3=wmaxlev(s,w)  
% 它和下面这个函数一样  
l4=wmaxlev(min(s),w)  
% 改变小波  
w='db7';  
% 计算最大分解尺度  
l5=wmaxlev(s,w)
```

程序运行结果如下：

```
l1 =  
    10  
l2 =  
     6  
l3 =  
     7  
l4 =  
     7  
l5 =  
     3
```

### 16.1.2 信号的离散小波分解

MATLAB7 提供了一维离散小波分解函数，下面对这些函数进行详细的介绍。

#### 1. 多尺度一维小波分解函数

在 MATLAB 中实现多尺度一维小波分解的函数是 `wavedec`，其调用格式有以下两种：

```
[C,L]=wavedec(X,N,'wname')  
[C,L]=wavedec(X,N,Lo_D,Hi_D)
```

`wavedec` 使用给定的小波 ('wname') 或者小波滤波器 (`Lo_D` 和 `Hi_D`) 执行一维多尺度小波分析。

格式 返回信号  $X$  在  $N$  层的小波分解。 $N$  必须是严格的正整数。输出分解结构包含小波解向量  $C$  和相应的记录向量  $L$ 。

格式 使用指定的低通和高通分解滤波器，返回分解结构。

算法分析：给定长度为  $N$  的信号，那么 DWT 顶多包含  $\log_2 N$  层。第一步由信号  $s$  产生两个系数集：低频系数  $cA_1$  和高频系数  $cD_1$ 。 $cA_1$  由  $s$  和低通滤波器  $Lo\_D$  卷积获得， $cD_1$  由  $s$  和高通滤波器卷积获得。

每个滤波器的长度为  $2N$ ，如果  $n=\text{length}(s)$ ，则信号  $F$  和  $G$  的长度为  $n+2N-1$ ，系数  $cA_1$  和  $cD_1$  的长度为  $\text{floor}(\frac{n-1}{2}) + N$ 。

下一步分解就是使用同样的框架将低频系数  $cA_1$  分解成两部分，即  $cA_1$  替换为  $cA_2$  和  $cD_2$ ，依次类推。

信号  $s$  在第  $j$  层的分解结构为  $[cA_j, cD_j, \dots, cD_1]$ 。

【例 16-7】wavedec 函数应用举例。

```
% 当前延拓模式是补零
% 装载一维原始信号
load sumsin;s=sumsin
subplot(211);plot(s)
title('原始信号');
% 使用 db1 进行 3 层分解
[c,l]=wavedec(s,3,'db1');
subplot(212);plot(c);title('小波分解结构');
xlabel('低频系数和第 3~1 层的高频系数');
```

程序运行的结果如图 16-8 所示。

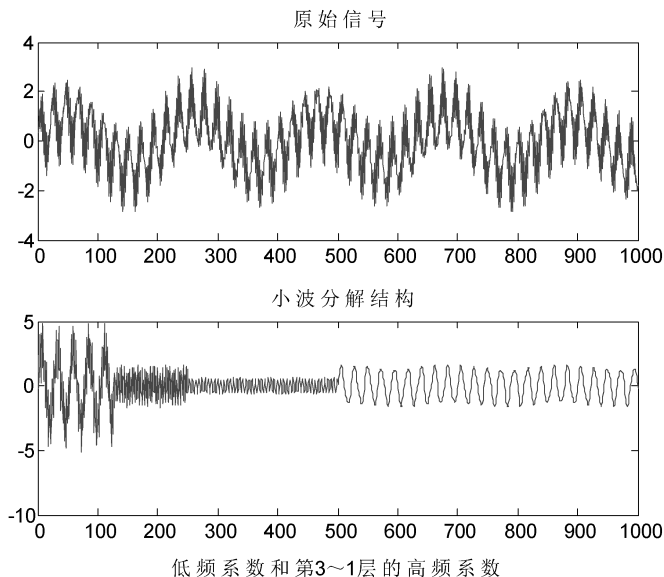


图 16-8 小波分解结构



## 2. 单尺度一维离散小波分解函数

在 MATLAB 中实现单尺度一维离散小波变换的函数是 `dwt`，其调用格式有以下 4 种：

```
[cA,cD]=dwt(X,'wname')
[cA,cD]=dwt(X,'wname','mode',MODE)
[cA,cD]=dwt(X,Lo_D,Hi_D)
[cA,cD]=dwt(X,Lo_D,Hi_D,'mode',MODE)
```

`dwt` 命令使用特定小波（'wname'）或者特定的小波分解滤波器（`Lo_D` 和 `Hi_D`）执行单层一维小波分解。

格式 计算低频系数向量 `cA` 和低频系数向量 `cD`，由向量 `X` 进行小波分解得到。字符串 'wname' 包含小波名。

格式 和上面一样计算小波分解，`Lo_D` 和 `Hi_D` 滤波器为输入。其中，`Lo_D` 是分解低频滤波器，`Hi_D` 是分解高通滤波器。这两个滤波器必须具有相同的长度。

令 `lx` 为 `X` 的长度，`lf` 为滤波器 `Lo_D` 和 `Hi_D` 的长度，如果 DWT 扩展模式设置为 `periodization`，`length(cA)=length(cD)=la`，这里 `la=ceil(lx/2)`，那么对于其他扩展模式，`la=floor(lx+lf-1)/2`。

格式 `[cA,cD]=dwt(..., 'mode',MODE)` 使用指定的 `MODE` 扩展模式计算小波分解。例如，`[cA,cD]=dwt(x,'db1','mode','sym');`

### 【例 16-8】dwt 函数应用举例。

```
% 当前扩展模式是补零模式
% 构造原始一维尺度信号
randn('seed',531316785)
s=2+kron(ones(1,8),[1,-1])+((1:16).^2)/32+0.2*randn(1,16);
% 进行单尺度离散 haar 小波变换
[ca1,cd1]=dwt(s,'haar');
subplot(311);plot(s);title('原始信号');
subplot(323);plot(ca1);title('haar 低频系数');
subplot(324);plot(cd1);title('haar 高频系数');
% 对于给定的小波，计算两个相关的分解滤波器，并直接使用该滤波器计算低频和高频系数
[Lo_D,Hi_D]=wfilters('haar','d');
[ca1,cd1]=dwt(s,Lo_D,Hi_D);
% 进行单尺度 db2 离散小波变换并观察最后系数的边缘效果
[ca2,cd2]=dwt(s,'db2');
subplot(325);plot(ca2);title('haar 低频系数');
subplot(326);plot(cd2);title('haar 高频系数');
```

程序运行结果如图 16-9 所示。

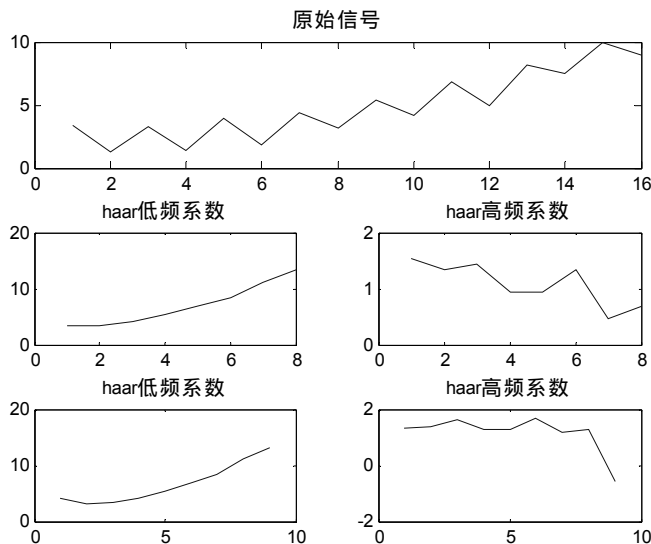


图 16-9 dwt 分解结果

### 3. 离散小波变换扩展模式函数

在 MATLAB 中实现离散小波变换扩展模式的函数是 `dwtmode`，其调用格式有如下两种：

```
ST=dwtmode
dwtmode('mode')
```

`dwtmode` 命令用来设置信号或者图像进行离散小波和小波包变换的扩展模式。扩展模式表示信号或者图像分析时边界问题的处理方法。

`dwtmode` 或者 `dwtmode('status')` 显示当前模式。

`ST=dwtmode` 或者 `ST=dwtmode('status')` 显示并返回当前模式，由变量 `ST` 表示。

`ST=dwtmode('status','nodisp')` 返回当前模式，由变量 `ST` 表示，但 MATLAB 命令窗口中无显示信息。

`dwtmode('per')` 设置 DWT 模式为周期延拓。这种模式产生最短长度的小波分解，但是用于 IDWT 的扩展模式一定要一样，以保证重构的效果。

使用这种模式，`dwt` 和 `dwt2` 产生与以前版本的函数 `dwptper` 和 `dwtper2` 一样的结果。

所有的函数和 GUI 工具包括一维、二维 DWT 以及一维、二维小波包变换使用指定的 DWT 扩展模式。

`dwtmode` 可以更新全局变量，使用这 6 种信号延拓模式。这些延拓模式必须通过该函数进行改变，避免直接更改全局变量。

默认的模式从当前路径下的文件 `DWTMODE.DEF` 载入。如果该文件不存在，则可使用 `DWTMODE.CFG` 文件（路径为 `toolbox/wavelet/wavelet`）。

`dwtmode('save',MODE)` 将 `MODE` 作为一种默认模式存储到 `DWTMODE.DEF` 文件，如果当前目录下该文件已经存在，那么先删除然后再存储。

```
dwtmode('save') 等同于 dwtmode('save',CURRENTMODE)
```



## 【例 16-9】dwtmode 函数应用举例。

```
% 如果 DWT 延拓模式全局变量不存在，默认的是 Symmetrization
clear global
dwtmode

-----
** DWT Extension Mode: Symmetrization (half-point) **
-----

% 显示当前 DWT 信号延拓模式
dwtmode

-----
** DWT Extension Mode: Symmetrization (half-point) **
% 改为周期延拓模式
dwtmode('per')
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! WARNING: Change DWT Extension Mode !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
-----

** DWT Extension Mode: Periodization **
% 显示当前的 DWT 信息
dwtmode

-----

** DWT Extension Mode: Periodization **
-----
```

## 16.2 信号重构

### 16.2.1 信号小波重构

#### 1. 单尺度一维小波逆变换函数

在 MATLAB 中实现单尺度一维小波逆变换的函数是 `idwt`，其调用格式有以下 5 种：

```
X=idwt(cA,cD,'wname')
X=idwt(cA,cD,Lo_R,Hi_R)
X=idwt(cA,cD,'wname',L)
X=idwt(cA,cD,Lo_R,Hi_R,L)
X=idwt(...,'mode',MODE)
```

`idwt` 命令使用指定的小波（'wname'）或者小波重构滤波器（`Lo_R` 和 `Hi_R`）进行单尺度一维重构。

格式 返回使用小波'wname'，返回单尺度重构的低频系数向量 `X` 及高频系数向量 `cA` 和 `cD`。

格式 使用的是低通滤波器  $Lo\_R$  和重构高通滤波器  $Hi\_R$ 。这两个滤波器必须有同样的长度。

令  $la$  为  $cA$  的长度 (即  $cD$  的长度),  $lf$  为滤波器  $Lo\_R$  和  $Hi\_R$  的长度, 如果 DWT 扩展模式设为 `periodization`, 那么  $length(X)=LX$ , 这里  $LX=2la$ 。对于其他扩展模式, 那么  $LX=2la-lf+2$ 。

格式 和格式 返回由 `idwt(cA,cD,'wname')` 得到的长度为  $L$  的中间部分。 $L$  必须小于  $LX$ 。

格式 使用指定的延拓模式 `MODE` 进行小波重构。

格式  $X=idwt(cA,[],...)$  基于低频系数向量  $cA$ , 返回单尺度重构低频系数  $X$ 。

格式  $X=idwt([],cD,...)$  基于高频系数向量  $cD$ , 返回单尺度重构高频系数  $X$ 。

由  $j$  层的低频和高频系数  $cA_j$  和  $cD_j$  开始, 离散小波逆变换采用和分解相反的步骤, 插入零并将结果和重构滤波器卷积, 重构出  $cA_{j-1}$ 。

【例 16-10】`idwt` 函数应用举例。

```
% 当前扩展模式是补零
% 构造原始一维信号 s
randn('seed',531316785);
s=2+kron(ones(1,8),[1 -1])+((1:16).^2)/32+0.2*randn(1,16);
% 使用 db2 进行单尺度 dwt
[ca1,cd1]=dwt(s,'db2');
subplot(221);plot(ca1);
title('db2 低频系数');
subplot(222);plot(cd1);
title('db2 高频系数');
% 进行单尺度离散小波逆变换
ss=idwt(ca1,cd1,'db2');
err=norm(s-ss); % 检查重构
subplot(212);plot([s;ss]);
title('原始信号和重构信号');
xlabel(['误差的 2 范数为',num2str(err)])
% 对于给定的小波, 计算两个相关重构滤波器, 并直接利用它们进行逆变换
[Lo_R,Hi_R]=wfilters('db2','r');
ss=idwt(ca1,cd1,Lo_R,Hi_R);
```

程序运行结果如图 16-10 所示。

## 2. 多尺度一维小波重构函数

在 MATLAB 中实现多尺度一维小波重构的函数是 `waverec`, 其调用格式有以下两种:

```
X=waverec(C,L,'wname')
X=waverec(C,L,Lo_R,Hi_R)
```

说明: `waverec` 使用指定的小波 ('wname') 或者小波重构滤波器 ( $Lo\_R$  和  $Hi\_R$ ) 进行一维多尺度小波重构。它返回的是原信号  $X$ , 即

$$X=waverec(X=N,wname,'wname')$$

因此可以看成是 `waverec` 的逆函数。



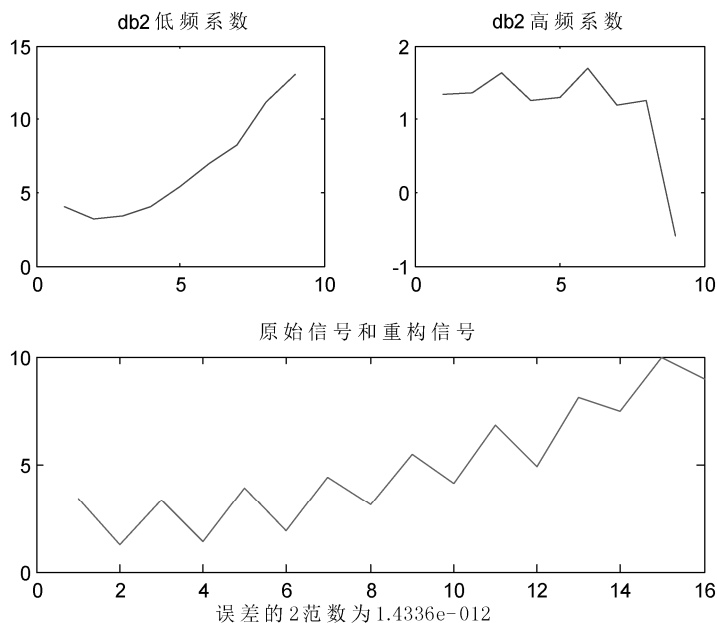


图 16-10 一维离散小波逆变换结果

格式 基于多尺度小波分解结构[C,L]和小波'wname'重构信号 X。

格式 使用指定的重构滤波器重构 X。其中，Lo\_R 是重构低通滤波器，Hi\_R 为重构高通滤波器。

`X=waverec(C,L,'wname')` 等同于 `X=appcoef(C,L,'wname',0)`

#### 【例 16-11】waverec 函数应用举例。

```
% 当前延拓模式是补零
% 装载一维原始
load leleccum;s=leleccum(1:3920);ls=length(s);
% 使用 db5 进行尺度为 3 时的分解
[c,l]=wavedec(s,3,'db5');
% 从小波分解结构[c,l]重构信号 s
a0=waverec(c,l,'db5');
% 检查重构效果
err=norm(s-a0);
subplot(2,1,1);plot(s);title('原始信号')
subplot(2,1,2);plot(a0);title('重构信号')
```

程序运行的结果如图 16-11 所示。

计算结果如下：

```
err =
    1.6717e-009
```

可以看出，重构信号和原信号相比，失真非常小。

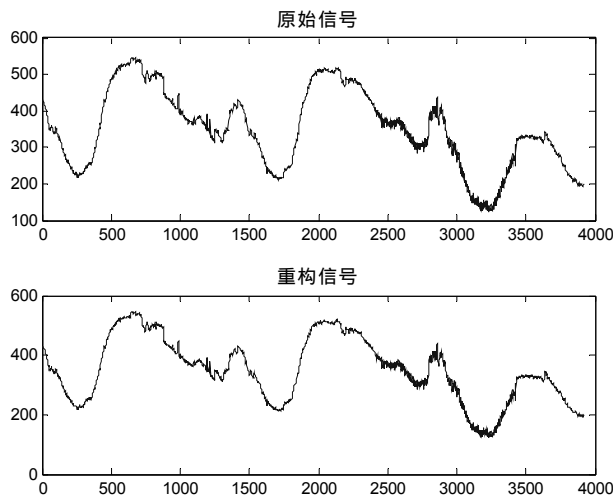


图 16-11 多尺度一维小波重构结果

### 3. 一维小波单支重构函数

在 MATLAB 中实现一维小波单支重构的函数是 `wrcoef`，其调用格式有以下 4 种：

```
X=wrcoef('type',C,L,'wname',N)
X=wrcoef('type',C,L,Lo_R,Hi_R,N)
X=wrcoef('type',C,L,'wname')
X=wrcoef('type',C,L,Lo_R,Hi_R,N)
```

`wrcoef` 基于小波分解结构  $[C,L]$ ，以及指定的小波（'wname'）或者重构滤波器（ $Lo\_R$  和  $Hi\_R$ ）来重构一维信号的系数。

**格式** 基于小波分解结构  $[C,L]$  在  $N$  层计算重构系数向量，'wname' 包含小波名字的字串。

变量 'type' 决定重构的系数是低频（'type'='a'）还是高频（'type'='d'）。当 'type'='a' 时， $N$  可以是 0；否则， $N$  必须是严格的正整数，且  $N \leq \text{length}(L)-2$ 。

**格式** 根据指定的重构滤波器计算系数。

**格式** 和 **格式** 重构系数的最大层数为  $N=\text{length}(L)-2$ 。

**【例 16-12】** `wrcoef` 函数应用举例。

```
% 采用补零的扩展模式
% 装载一维信号
load sumsin;s=sumsin;
subplot(211);plot(s);title('原始信号');
% 使用 sym4 进行尺度为 5 的分解
[c,l]=wavedec(s,5,'sym4');
% 从小波分解结构[C,L]，重构尺度为 5 的低频部分
a5=wrcoef('a',c,l,'sym4',5);
subplot(212);plot(a5);title('重构低频信号');
```



程序运行结果如图 16-12 所示。

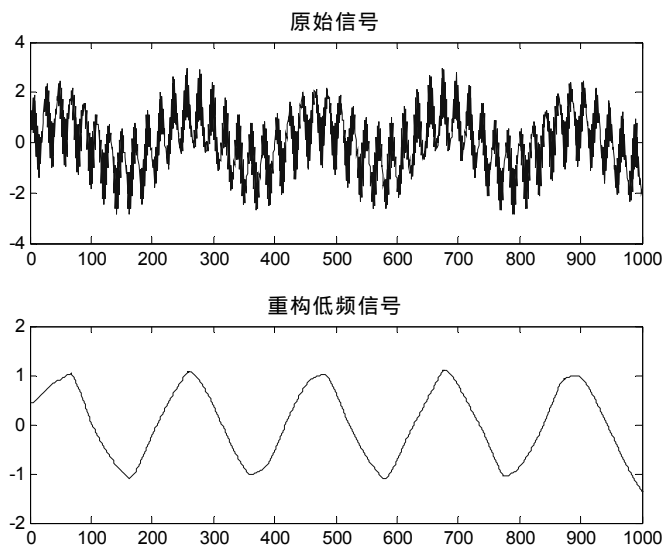


图 16-12 单支重构的结果

#### 4. 一维小波系数直接重构函数

在 MATLAB 中实现一维小波系数直接重构的函数是 `upwlev`，其调用格式有如下 6 种：

```
Y=upcoef(O,X,'wname',N)
Y=upcoef(O,X,'wname',N,L)
Y=upcoef(O,X,Lo_R,Hi_R,N)
Y=upcoef(O,X,Lo_R,Hi_R,N,L)
Y=upcoef(O,X,'wname')
Y=upcoef(O,X,Lo_R,Hi_R)
```

格式 计算向量  $X$  向上  $N$  步的重构系数。'wname'是包含小波名的字符串。 $N$  必须是严格的正整数。

如果  $O='a'$ ，则重构低频系数；如果  $O='d'$ ，则重构高频系数。

格式 计算向量  $X$  向上  $N$  步的重构系数，并取出结果中长度为  $L$  的中间部分。

如果不给定小波名，可以使用滤波器。格式 和格式 中， $Lo\_R$  是重构低通滤波器， $Hi\_R$  是重构高通滤波器。

```
Y=upcoef(O,X,'wname')等同于 Y=upcoef(O,X,'wname',1)。
Y=upcoef(O,X,Lo_R,Hi_R) 等同于 Y=upcoef(O,X,Lo_R,Hi_R,1)。
```

#### 【例 16-13】upcoef 函数应用举例 1。

```
% 当前扩展模式是补零
% 低频信号由 1~6 层系数获得
cfs=[1];
essup=10;
figure(1)
```

```

for i=1:6
    rec=upcoef('a',cfs,'db6',i);
    % essup 是重构信号必需的
    % 当 j 等于 essup 时, rec(j)非常小
    ax=subplot(6,1,i),h=plot(rec(1:essup));
    set(ax,'xlim',[1 325]);
    essup=essup*2;
end
subplot(6,1,1)
title(['尺度 1 ~ 6 , 由唯一的系数获得的低频信号'])

```

程序运行结果如图 16-13 所示。

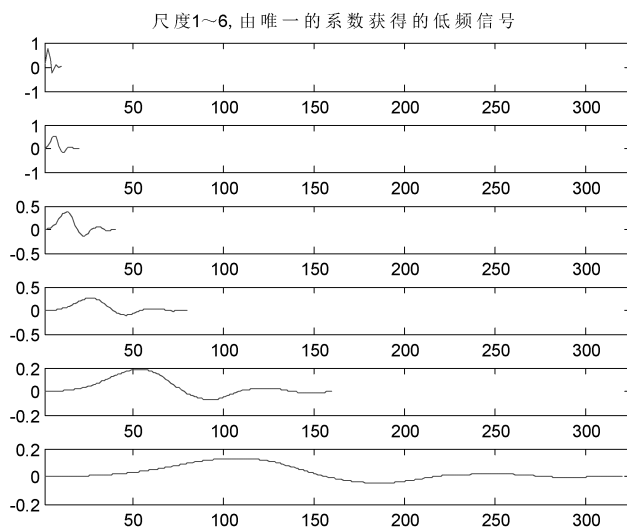


图 16-13 upcoef 函数应用举例 1

【例 16-14】upcoef 函数应用举例 2。

```

% 当前扩展模式是补零
% 低频信号由 1 ~ 6 层系数获得
cfs=[1];
mi=12;ma=30; % db6 小波滤波器是必需的
rec=upcoef('d',cfs,'db6',1);
figure(2)
subplot(6,1,1),plot(rec(mi*2^0:ma*2^0));
for i=2:6
    rec=upcoef('d',cfs,'db6',i);
    subplot(6,1,i),plot(rec(mi*2^(i-2):ma*2^(i-2)))
end
subplot(6,1,1)
title(['由唯一的系数从尺度 1 ~ 6 获得的低频信号'])

```



程序运行结果如图 16-14 所示。

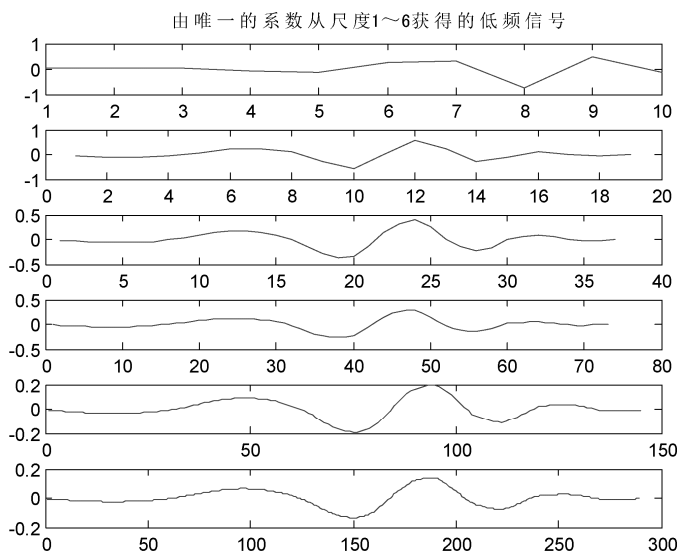


图 16-14 upcoef 函数应用举例 2

## 16.2.2 小波函数应用实例

在 MATLAB 中实现由模式构造小波的函数是 `pat2cwav`，其调用格式如下：

```
[PSI, XVAL, NC] = pat2cwav(YPAT, METHOD, POLDEGREE, REGULARITY)
```

该函数计算由 `XVAL` 和 `PSI` 给定并用于连续小波变换的小波函数，该小波向量 `YPAT` 定义的模式构造，方差为 1。其中，模式隐含的 `x` 值是 `xpat=linespace(0,1,length(YPAT))`。

常数 `NC` 的选取应保证通过以下方式的最小二乘拟合，`NC*PSI` 在区间 `[0,1]` 上近似于 `YPAT`。

当 `METHOD` 等于 `'polynomial'` 时，为 `POLDEGREE` 阶多项式；当 `METHOD` 等于 `'othconst'` 时，为正交函数空间的投影。

参数 `REGULARITY` 定义了 `0` 和 `1` 点的边界约束，可以是 `'continuous'`、`'differentiable'` 或 `'none'`。

当 `METHOD` 为 `'polynomial'` 时，若 `REGULARITY` 等于 `'continuous'`，则必须 `POLDEGREE` 3；若 `REGULARITY` 等于 `'differentiable'`，则必须 `POLDEGREE` 5。

**【例 16-15】**由给定模式产生一种新小波。

装载原始模式：正弦信号。

在命令窗口输入：

```
>> load ptpssin1;
>> whos
```

按 Enter 键，得到：

Name	Size	Bytes	Class
------	------	-------	-------

IntVAL	1x1	8	double array
X	1x256	2048	double array
Y	1x256	2048	double array
caption	1x35	70	char array
Grand total is 548 elements using 4174 bytes			

变量 X 和 Y 包含了模式。这个例子是一个演示实例，所以不但能得到模式的整体，而且能得到标题变量的重构细节。

在命令窗口输入：

```
>> IntVAL
```

按 Enter 键，得到：

```
ntVAL =  
0.1592
```

该模式定义在区间[0,1]上，积分值为 0.1592。它不是真正的小波，但看上去很相似。

在命令窗口输入：

```
>> plot(X,Y),title('原始模式')
```

按 Enter 键，得到如图 16-15 所示的图形。

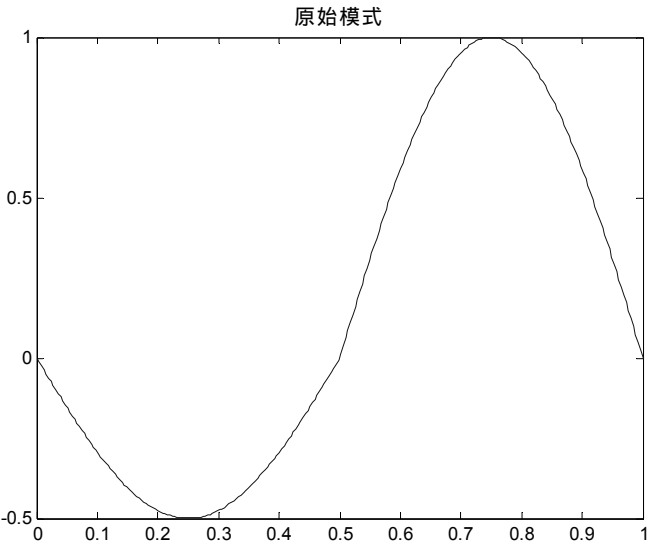


图 16-15 原始模式

为了合成适用于给定模式的新小波，我们使用 6 阶最小二乘多项式近似，并在开始和结束点采用连续性约束。

在命令窗口输入：

```
>> [psi,xval,nc]=pat2cwav(Y,'polynomial',6,'continuous');  
>> plot(X,Y,'-',xval,nc*psi,'--'),title('原始模式和构造的小波（虚线）')
```

按 Enter 键，得到图 16-16 所示的图形。新小波由 xval 和 nc\*psi 给定。

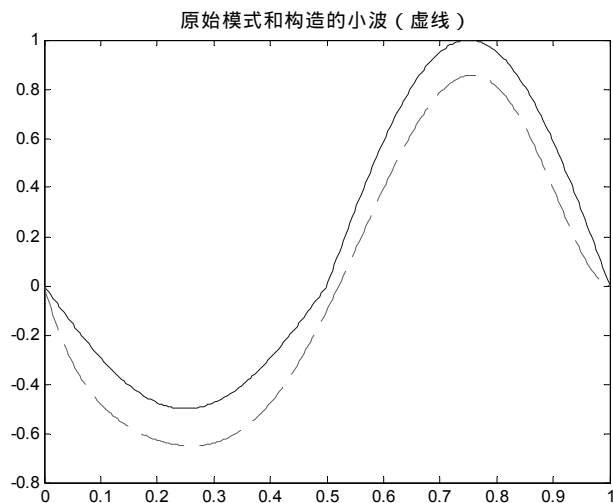


图 16-16 原始模式和构造的小波

【例 16-16】装载 vonkoch 信号并进行连续小波变换。

下面是装载 vonkoch 信号并进行连续小波变换的综合实例代码：

```
% 装载实际信号
load vonkoch;
vonkoch=vonkoch(1:510);
lv=length(vonkoch);

subplot(311),plot(vonkoch);title('被分析信号');
set(gca,'Xlim',[0 510])
% 执行离散 5 层 sym2 小波变换
% 层数 1~5 分别对应尺度 2、4、8、16 和 32
[c,l]=wavedec(vonkoch,5,'sym2');
% 扩展离散小波系数进行画图
% 层数 1~5 分别对应尺度 2、4、8、16 和 32
cfd=zeros(5,lv);
for k=1:5
    d=detcoef(c,l,k);
    d=d(ones(1,2^k),:);
    cfd(k,:)=wkeep(d(:),lv);
end

cfd=cfd(:);
I=find(abs(cfd)<sqrt(eps));
cfd(I)=zeros(size(I));
cfd=reshape(cfd,5,lv);

% 画出离散系数
subplot(312),colormap(pink(64));
img=image(flipud(wcodemat(cfd,64,'row')));
```



```
set(get(img,'parent'),'YtickLabel',[]);  
title('离散变换，系数绝对值');  
ylabel('层数');  
% 执行连续小波 sym2 变换，尺度从 1~32  
subplot(313);  
ccfs=cwt(vonkoch,1:32,'sym2','plot');  
title('离散变换，系数绝对值');  
colormap(pink(64));  
ylabel('尺度');
```

程序运行结果如图 16-17 ~ 图 16-19 所示。

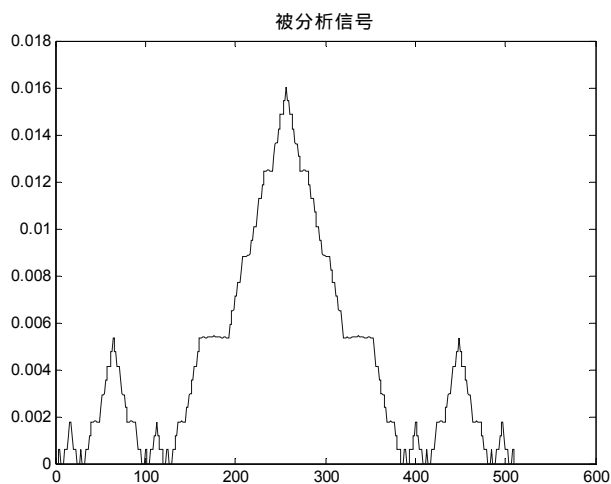


图 16-17 被分析信号

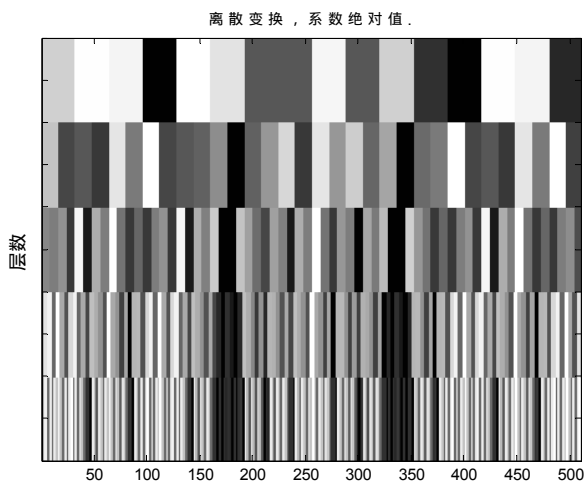
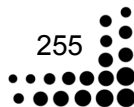


图 16-18 离散变换系数





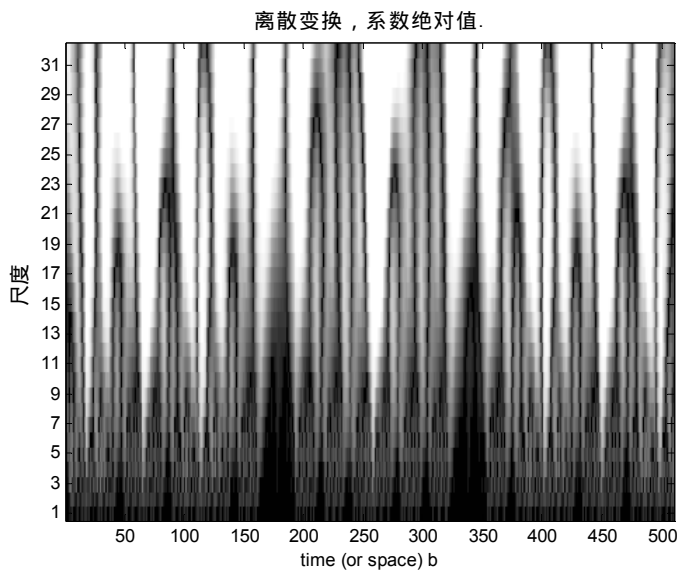


图 16-19 连续变换系数

下面介绍一维连续小波 MATLAB 命令行实现方法。小波工具箱只需要一个函数 `cwt` 来实现连续小波分析。

在 MATLAB 命令窗口中输入：

```
>> load leleccum;  
>> c=cwt(leleccum,1:32,'db4');  
>> c=cwt(leleccum,1:48,'db4','plot');
```

显示结果如图 16-20 所示。

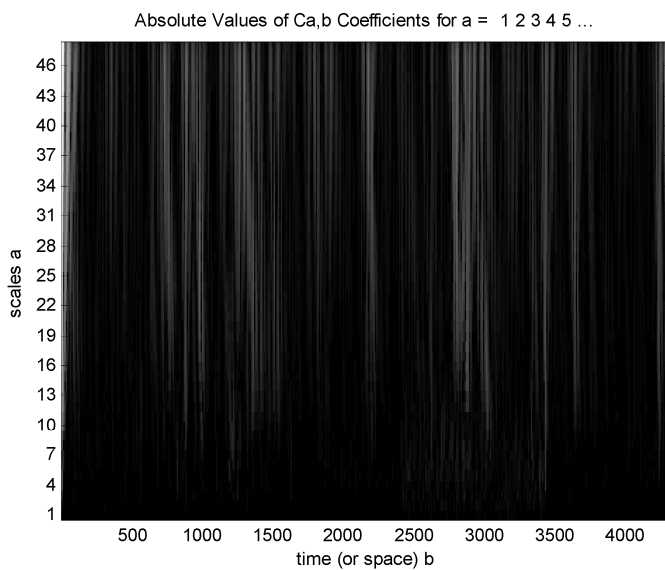


图 16-20 系数的图形表示

通过命令行中的 whos 命令可查看工作区的 leleccum 信号。其中第二条指令执行一维小波变换，c 中的每行分别对应于某个尺度变换后的系数。产生的系数图形如图 16-21 所示。

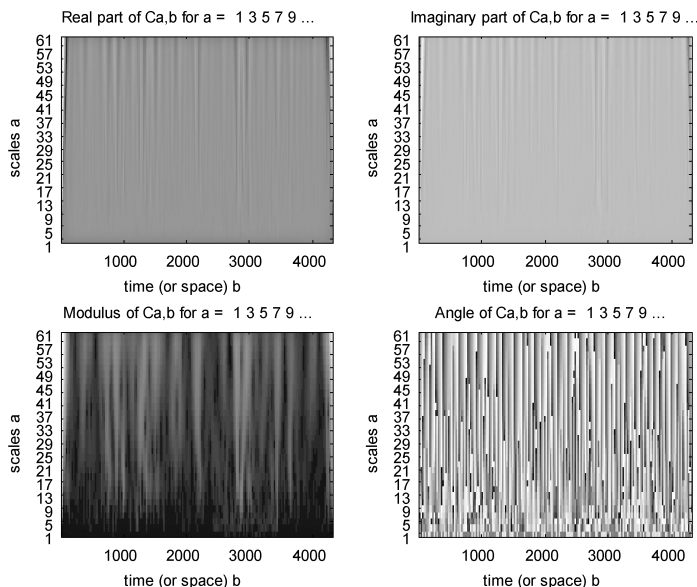


图 16-21 复小波变换得到的系数

重新选择分析尺度，在命令行窗口输入：

```
>>c=cwt(leleccum,2:2:128,'db4','plot');
```

重新选择分析尺度后，指令运行结果如图 16-22 所示。

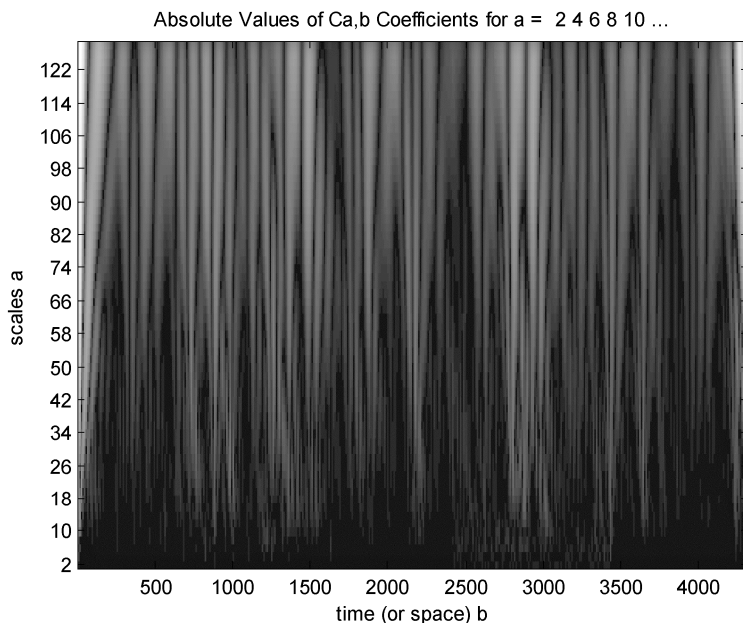


图 16-22 重新选择分析尺度



下面给出一个连续复小波变换的例子。

(1) 装载信号。在 MATLAB 命令行窗口中输入：

```
>> load leleccum;
```

该信号定义为： $x=\text{linspace}(0,1,1024); y=\exp(-128*((x-0.3).^2))-3*(\text{abs}(x-0.7).^0.4)$ 。

(2) 完成连续小波变换。在 MATLAB 命令窗口中输入：

```
>> c=cwt(leleccum,1:2:64,'cgau4');
```

这里的复小波函数为 cgau4。小波工具箱中共有 4 种复小波，即 cgau、shan、fbsp 和 cmor。可以通过 waveinfo 命令来查看其主要性质。

(3) 显示系数的图形表示。函数 cwt 的第 4 个参数可以产生复小波分析的 4 个变换系数实部和虚部、模和相位，这是与 6.1 节中实小波所不同的地方。输入以下命令：

```
>> c=cwt(leleccum,1:2:64,'cgau4','plot');
```

则相应的系数表示见图 16-21，分别为实部和虚部、模和相位。

【例 16-17】利用小波分解分析第一段电力载波信号数据成分。

```
%装载采集的信号 leleccum.mat
load leleccum;
%将信号中第 3600~3700 个采样点赋给 s
index=3600:3700;
s=leleccum(index);
%画出原始信号
figure(1);
plot(index,s);
ylabel('幅值 A');
xlabel('样本序号 n');
%用 db3 小波进行 5 层分解
[c,l]=wavedec(s,5,'db3');
%重构第 1~5 层逼近系数
a5=wrcoef('a',c,l,'db3',5);
a4=wrcoef('a',c,l,'db3',4);
a3=wrcoef('a',c,l,'db3',3);
a2=wrcoef('a',c,l,'db3',2);
a1=wrcoef('a',c,l,'db3',1);
%显示逼近系数
figure(2)
subplot(5,2,1);
plot(index,a5,'LineWidth',2);
ylabel('a5');
subplot(5,2,3);
plot(index,a4,'LineWidth',2);
ylabel('a4');
```

```

subplot(5,2,5);
plot(index,a3,'LineWidth',2);
ylabel('a3');
subplot(5,2,7);
plot(index,a2,'LineWidth',2);
ylabel('a2');
subplot(5,2,9);
plot(index,a1,'LineWidth',2);
ylabel('a1');
xlabel('样本序号 n');
%重构第 1 ~ 5 层细节系数
d5 = wrcoef('d',c,l,'db3',5);
d4 = wrcoef('d',c,l,'db3',4);
d3 = wrcoef('d',c,l,'db3',3);
d2 = wrcoef('d',c,l,'db3',2);
d1 = wrcoef('d',c,l,'db3',1);
%显示细节系数
subplot(5,2,2);
plot(index,d5,'LineWidth',2);
ylabel('d5');
subplot(5,2,4);
plot(index,d4,'LineWidth',2);
ylabel('d4');
subplot(5,2,6);
plot(index,d3,'LineWidth',2);
ylabel('d3');
subplot(5,2,8);
plot(index,d2,'LineWidth',2);
ylabel('d2');
subplot(5,2,10);
plot(index,d1,'LineWidth',2);
ylabel('d1');
xlabel('样本序号 n');

```

第一段电力载波信号如图 16-23 所示，利用 db3 小波对其进行 5 层小波分解，得到的逼近信号和细节信号如图 16-24 所示。可以看出，细节信号 d1 和 d2 的值较小，可以认为是由传感器和状态噪声的高频分量引起的局部的干扰；细节信号 d4 包含了 3 个相连的主要信号模式，它最接近于原始数据的曲线；而细节信号 db5 含有的信息不多，因此第 4 层贡献最大，它提取了原始数据曲线的形状。

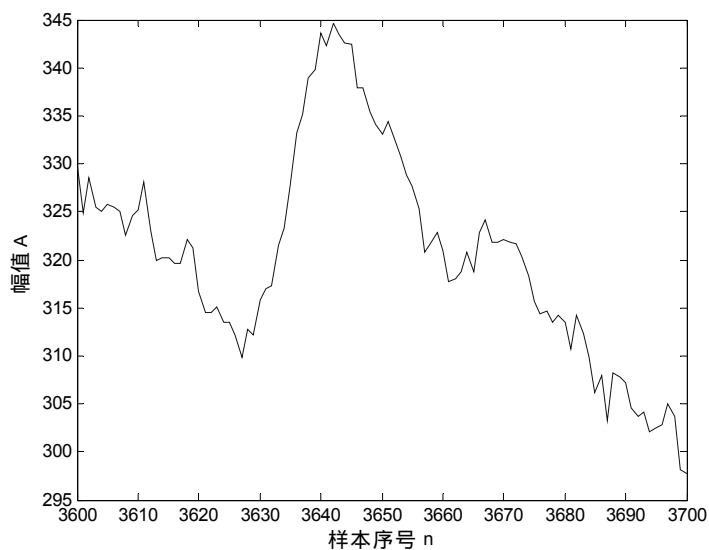


图 16-23 第一段数据波形

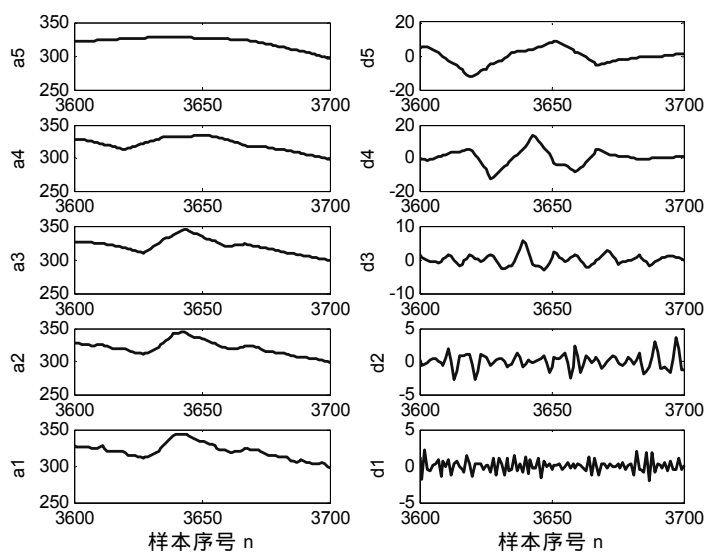


图 16-24 第一段数据小波分解结果

# 第 17 章 小波包在时频分析 案例中的应用

小波包在时频分析中具有明显的优点，下面给出几个典型信号分别在小波基及小波包的最优基下展开后的相平面情况。

## 17.1 小波包变换分析两个信号功率谱

【例 17-1】小波包变换分析两个信号功率谱。

下面是使用小波包变换分析两个信号的特征向量和各频率成分的功率谱。

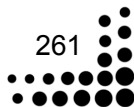
```
%t=0.001:0.001:1;
t=1:1000;
s1=sin(2*pi*50*t*0.001)+sin(2*pi*120*t*0.001)+rand(1,length(t));

for t=1:500;
s2(t)=sin(2*pi*50*t*0.001)+sin(2*pi*120*t*0.001)+rand(1,length(t));
end

for t=501:1000;
s2(t)=sin(2*pi*200*t*0.001)+sin(2*pi*120*t*0.001)+rand(1,length(t));
end

subplot(9,2,1)
plot(s1)
title('原始信号')
ylabel('S1')

subplot(9,2,2)
plot(s2)
title('故障信号')
ylabel('S2')
```





```
wpt=wpdec(s1,3,'db1','shannon');

%plot(wpt);
s130=wprcoef(wpt,[3,0]);
s131=wprcoef(wpt,[3,1]);
s132=wprcoef(wpt,[3,2]);
s133=wprcoef(wpt,[3,3]);
s134=wprcoef(wpt,[3,4]);
s135=wprcoef(wpt,[3,5]);
s136=wprcoef(wpt,[3,6]);
s137=wprcoef(wpt,[3,7]);
s10=norm(s130);
s11=norm(s131);
s12=norm(s132);
s13=norm(s133);
s14=norm(s134);
s15=norm(s135);
s16=norm(s136);
s17=norm(s137);

st10=std(s130);
st11=std(s131);
st12=std(s132);
st13=std(s133);
st14=std(s134);
st15=std(s135);
st16=std(s136);
st17=std(s137);

disp('正常信号的特征向量');
snorm1=[s10,s11,s12,s13,s14,s15,s16,s17]
std1=[st10,st11,st12,st13,st14,st15,st16,st17]

subplot(9,2,3);plot(s130);
ylabel('S130');
subplot(9,2,5);plot(s131);
ylabel('S131');
subplot(9,2,7);plot(s132);
ylabel('S132');
subplot(9,2,9);plot(s133);
ylabel('S133');
subplot(9,2,11);plot(s134);
```



```
ylabel('S134');
subplot(9,2,13);plot(s135);
ylabel('S135');
subplot(9,2,15);plot(s136);
ylabel('S136');
subplot(9,2,17);plot(s137);
ylabel('S137');

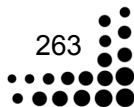
wpt=wpdec(s2,3,'db1','shannon');

%plot(wpt);
s230=wprcoef(wpt,[3,0]);
s231=wprcoef(wpt,[3,1]);
s232=wprcoef(wpt,[3,2]);
s233=wprcoef(wpt,[3,3]);
s234=wprcoef(wpt,[3,4]);
s235=wprcoef(wpt,[3,5]);
s236=wprcoef(wpt,[3,6]);
s237=wprcoef(wpt,[3,7]);

s20=norm(s230);
s21=norm(s231);
s22=norm(s232);
s23=norm(s233);
s24=norm(s234);
s25=norm(s235);
s26=norm(s236);
s27=norm(s237);

st20=std(s230);
st21=std(s231);
st22=std(s232);
st23=std(s233);
st24=std(s234);
st25=std(s235);
st26=std(s236);
st27=std(s237);

disp('故障信号的特征向量');
snorm2=[s20,s21,s22,s23,s24,s25,s26,s27]
std2=[st20,st21,st22,st23,st24,st25,st26,st27]
```







```
subplot(9,2,4);plot(s230);
ylabel('S230');
subplot(9,2,6);plot(s231);
ylabel('S231');
subplot(9,2,8);plot(s232);
ylabel('S232');
subplot(9,2,10);plot(s233);
ylabel('S233');
subplot(9,2,12);plot(s234);
ylabel('S234');
subplot(9,2,14);plot(s235);
ylabel('S235');
subplot(9,2,16);plot(s236);
ylabel('S236');
subplot(9,2,18);plot(s237);
ylabel('S237');

%fft
figure
y1=fft(s1,1024);
py1=y1.*conj(y1)/1024;
y2=fft(s2,1024);
py2=y2.*conj(y2)/1024;

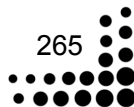
y130=fft(s130,1024);
py130=y130.*conj(y130)/1024;
y131=fft(s131,1024);
py131=y131.*conj(y131)/1024;
y132=fft(s132,1024);
py132=y132.*conj(y132)/1024;
y133=fft(s133,1024);
py133=y133.*conj(y133)/1024;
y134=fft(s134,1024);
py134=y134.*conj(y134)/1024;
y135=fft(s135,1024);
py135=y135.*conj(y135)/1024;
y136=fft(s136,1024);
py136=y136.*conj(y136)/1024;
y137=fft(s137,1024);
py137=y137.*conj(y137)/1024;

y230=fft(s230,1024);
```



```
py230=y230.*conj(y230)/1024;  
y231=fft(s231,1024);  
py231=y231.*conj(y231)/1024;  
y232=fft(s232,1024);  
py232=y232.*conj(y232)/1024;  
y233=fft(s233,1024);  
py233=y233.*conj(y233)/1024;  
y234=fft(s234,1024);  
py234=y234.*conj(y234)/1024;  
y235=fft(s235,1024);  
py235=y235.*conj(y235)/1024;  
y236=fft(s236,1024);  
py236=y236.*conj(y236)/1024;  
y237=fft(s237,1024);  
py237=y237.*conj(y237)/1024;
```

```
f=1000*(0:511)/1024;  
subplot(1,2,1);  
plot(f,py1(1:512));  
ylabel('P1');  
title('原始信号的功率谱')  
subplot(1,2,2);  
plot(f,py2(1:512));  
ylabel('P2');  
title('故障信号的功率谱')  
figure  
subplot(4,2,1);  
plot(f,py130(1:512));  
ylabel('P130');  
title('S130 的功率谱')  
subplot(4,2,2);  
plot(f,py131(1:512));  
ylabel('P131');  
title('S131 的功率谱')  
subplot(4,2,3);  
plot(f,py132(1:512));  
ylabel('P132');  
subplot(4,2,4);  
plot(f,py133(1:512));  
ylabel('P133');  
subplot(4,2,5);  
plot(f,py134(1:512));
```





```
ylabel('P134');
subplot(4,2,6);
plot(f,py135(1:512));
ylabel('P135');
subplot(4,2,7);
plot(f,py136(1:512));
ylabel('P136');
subplot(4,2,8);
plot(f,py137(1:512));
ylabel('P137');

figure
subplot(4,2,1);
plot(f,py230(1:512));
ylabel('P230');
title('S230 的功率谱')
subplot(4,2,2);
plot(f,py231(1:512));
ylabel('P231');
title('S231 的功率谱')
subplot(4,2,3);
plot(f,py232(1:512));
ylabel('P232');
subplot(4,2,4);
plot(f,py233(1:512));
ylabel('P233');
subplot(4,2,5);
plot(f,py234(1:512));
ylabel('P234');
subplot(4,2,6);
plot(f,py235(1:512));
ylabel('P235');
subplot(4,2,7);
plot(f,py236(1:512));
ylabel('P236');
subplot(4,2,8);
plot(f,py237(1:512));
ylabel('P237');
figure
%plottree(wpt)
```

程序运行结果如图 17-1 所示。

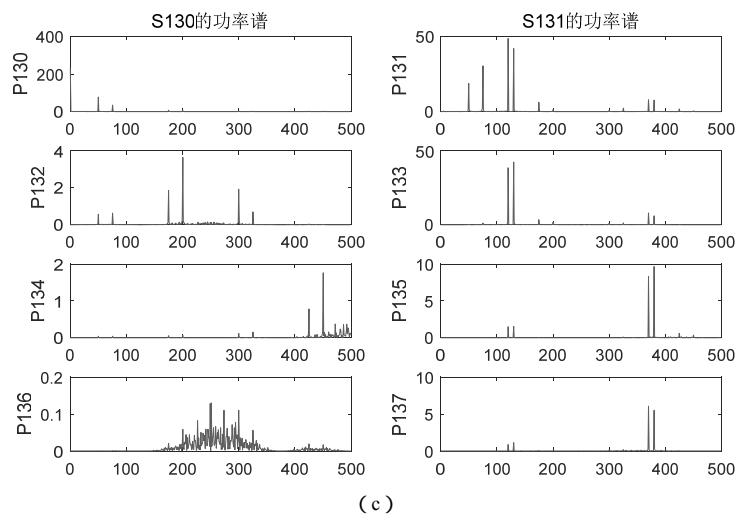
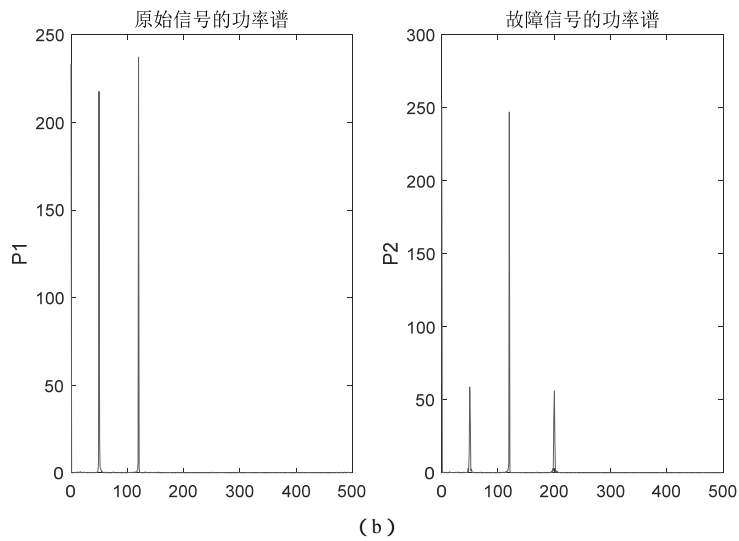
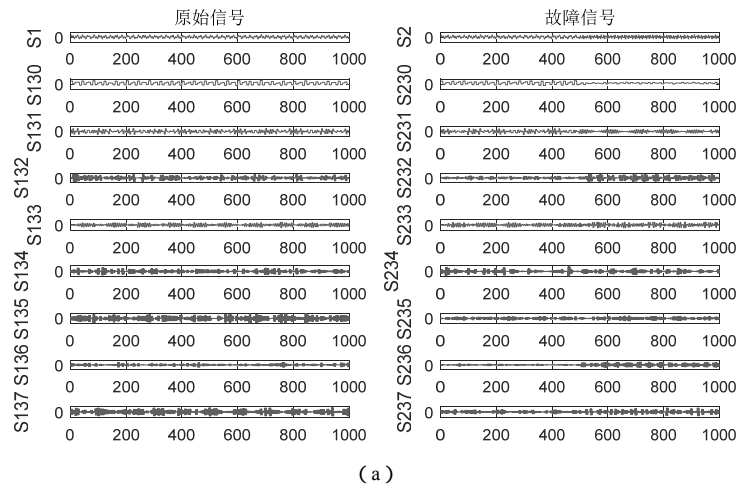


图 17-1 用小波包变换分析两个信号功率谱

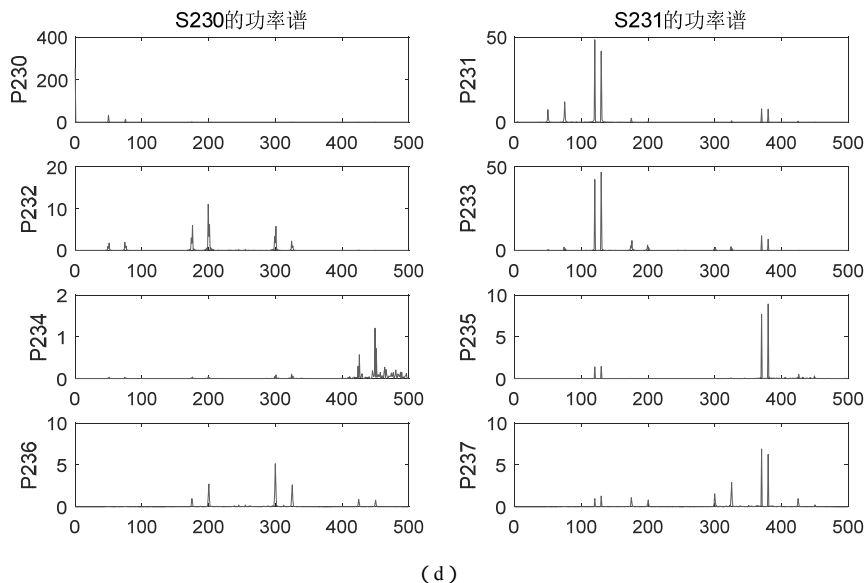


图 17-1 用小波包变换分析两个信号功率谱（续）

## 17.2 调频信号的小波包分析

调频信号的表达式为  $\sin(500\pi t + 20\sin 20\pi t)$ ，采样频率为 1000Hz。

【例 17-2】计算调频信号的小波包变换时频相平面。

```
%调频信号
for i=1:512
    x(i)=sin(500*pi*i/1000+20*sin(20*pi*i/1000));
end
figure(1);
plot(1:length(x),x);
xlabel('样本序号 n');
ylabel('幅值 A');
% 3 层 Haar 小波包
wpt = wpdec(x,3,'haar');
%最优基
T=besttree(wpt);
% 小波包结构
plot(wpt);
%haar 小波基下的相平面
wpviewcf(wpt,1);
```

变频信号的波形如图 17-2 所示，对其进行 3 层 Haar 小波的小波包分解，小波包分解结构如图 17-3 所示，分解后的时频相平面如图 17-4 所示。可见看出，频率与时间的关系为下一个正弦曲线，这与理论结果是一致的。

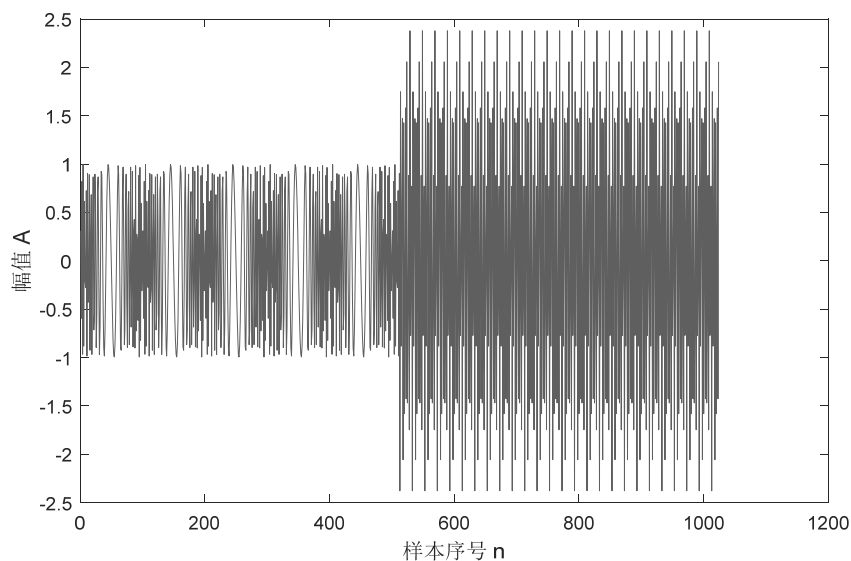


图 17-2 调频信号的时域波形

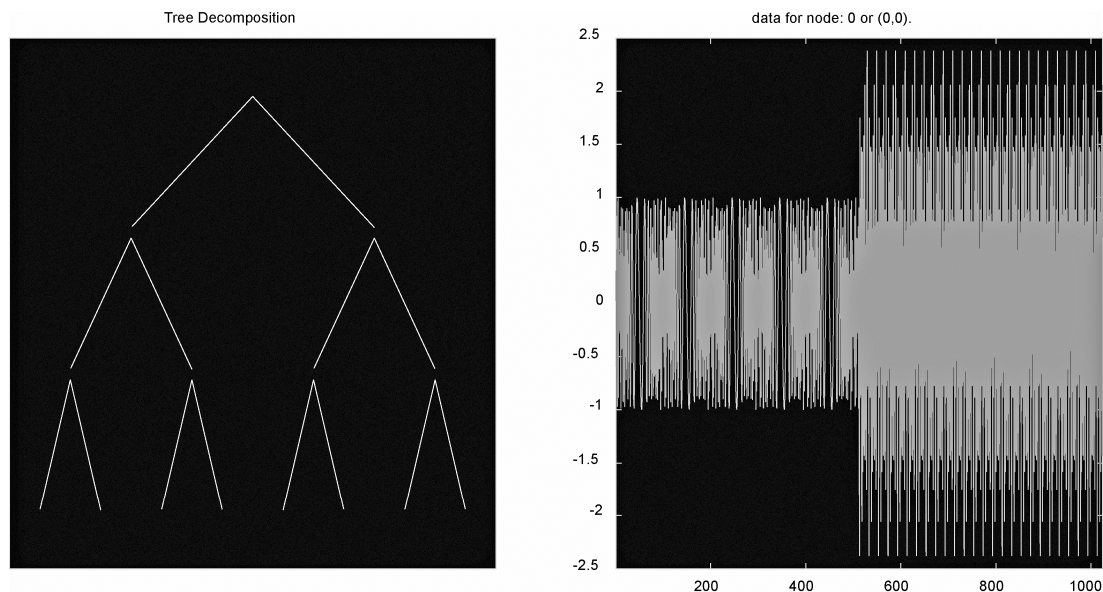


图 17-3 3 层 Haar 小波包分解结构

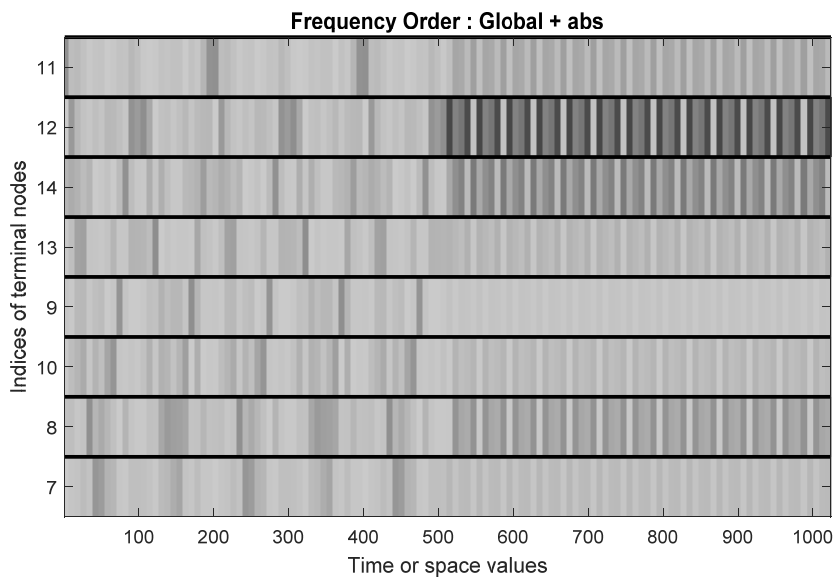


图 17-4 Haar 小波包分解的时频相平面

## 17.3 正弦信号的小波包分析

【例 17-3】计算正弦信号小波包变换的时频相平面。

```
%读正弦信号
load sinper8;
x = sinper8;
figure(1);
plot(1:length(x),x);
xlabel('样本序号 n');
ylabel('幅值 A');
% 7 层 Harr 小波包
% Shannon 熵.
wpt = wpdec(x,7,'haar','shannon');
%最优基
T=besttree(wpt);
% 小波包结构
plot(wpt);
%harr 小波基下的相平面
wpviewcf(wpt,1);
%最优基下的相平面
wpviewcf(T,1);
```

正弦信号的波形如图 17-5 所示，周期为 8，样本长度为 256。对其进行 7 层 Haar 小波的小波包分解，树结构如图 17-6 所示。

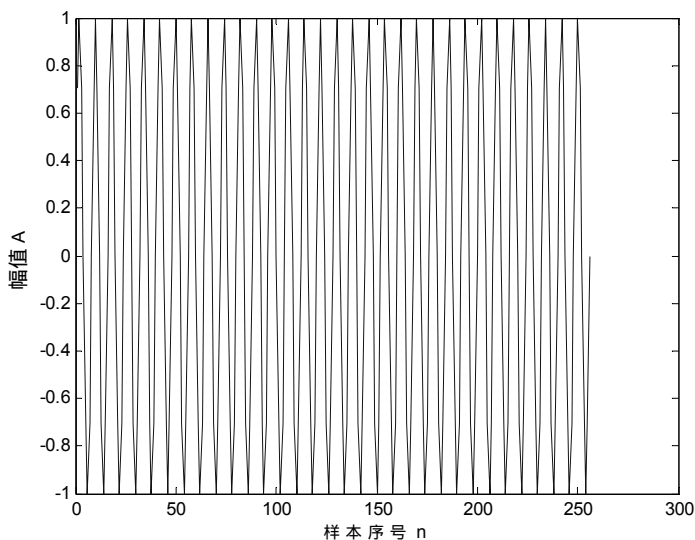


图 17-5 正弦信号波形

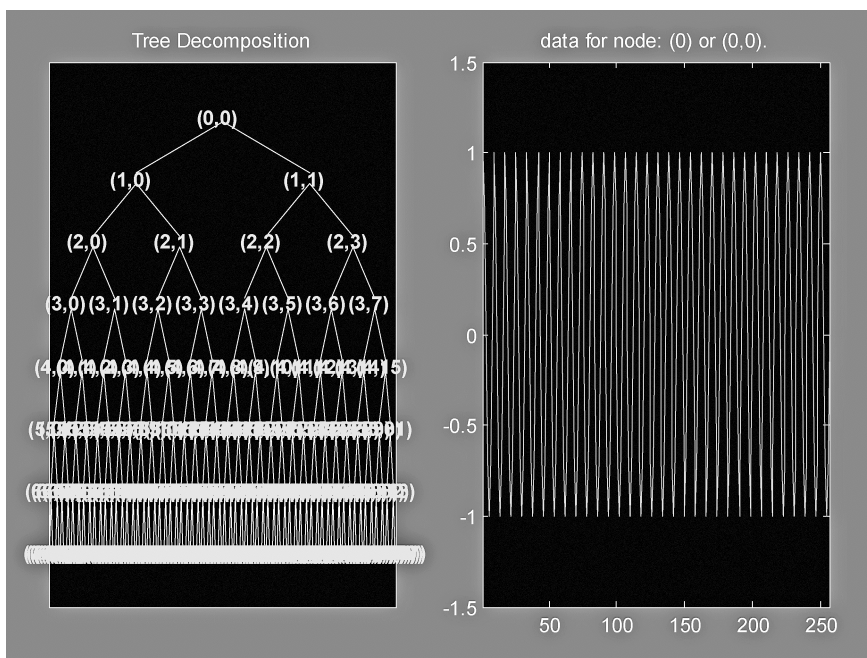


图 17-6 7 层 Haar 小波包分解的树结构

Harr 小波分解后的时频相平面如图 17-7 所示，最优基分解后的时频相平面如图 17-8 所示。可以看出，其频率是一条直线。



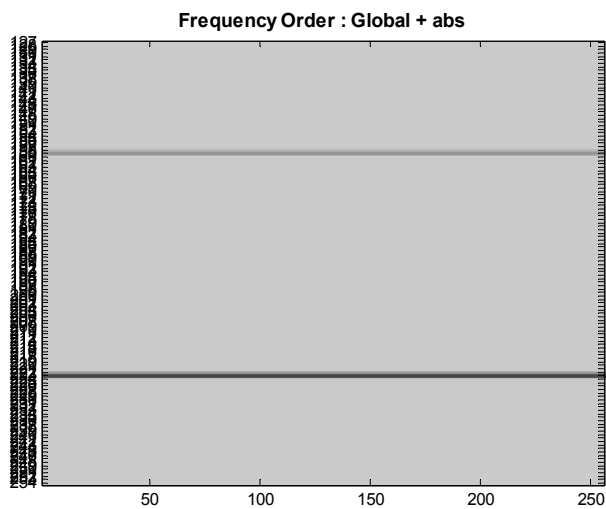


图 17-7 Haar 小波分解后的时频相平面

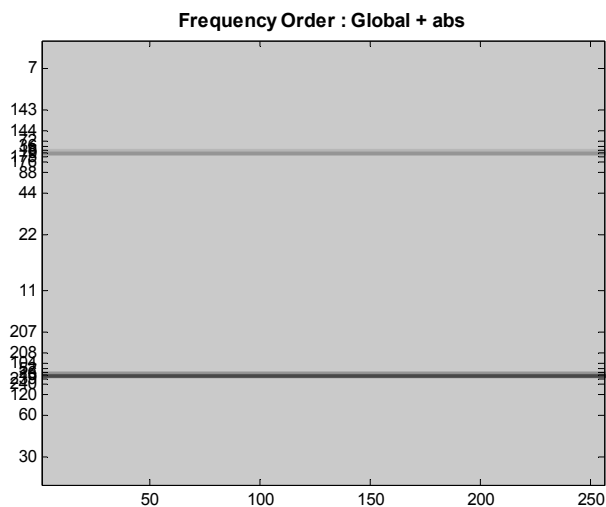


图 17-8 最优基分解后的时频相平面

## 17.4 $\delta$ 信号的小波包分析

【例 17-4】计算  $\delta$  信号小波包变换的时频相平面。

```
%delta 信号  
x=zeros(1,256);  
x(150)=1;  
figure(1);  
plot(1:length(x),x);  
xlabel('样本序号 n');
```



```

ylabel('幅值 A');
% 3 层 db1 小波包
wpt = wpdec(x,3,'db1');
%最优基
T=besttree(wpt);
% 小波包结构
plot(wpt);
%db1 小波基下的时频相平面
wpviewcf(wpt,1);

```

生成的  $\delta$  信号波形如图 17-9 所示，它在样本点 150 处存在一个单位冲激。对其进行 3 层 db1 小波的小波包分解，树结构如图 17-10 所示。

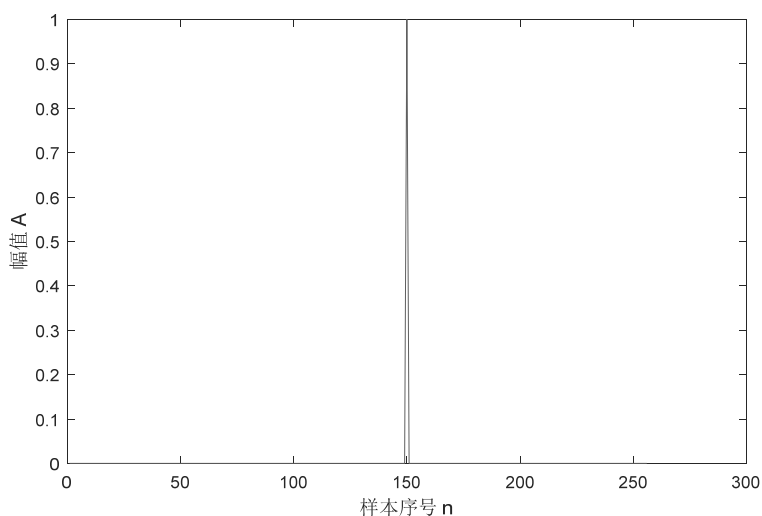


图 17-9  $\delta$  信号波形

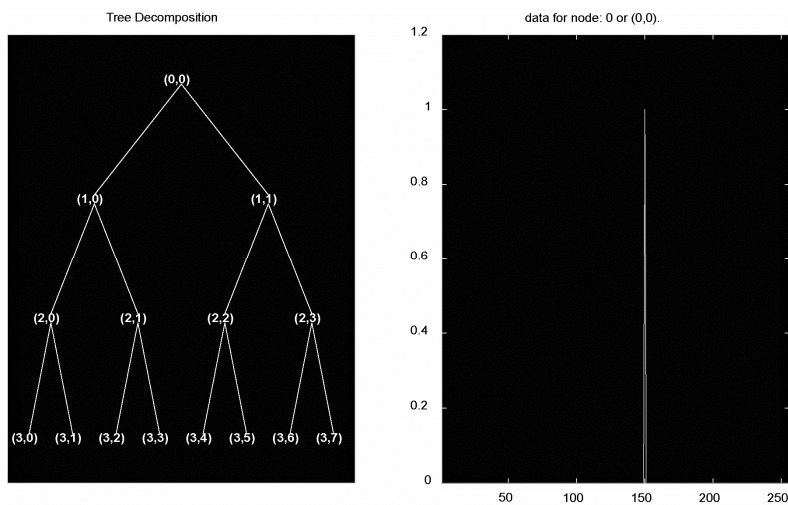
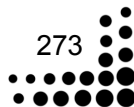


图 17-10 3 层 db1 小波包分解的树结构





分解后的相平面如图 17-11 所示，可见能够正确地确定冲击脉冲位置信息。

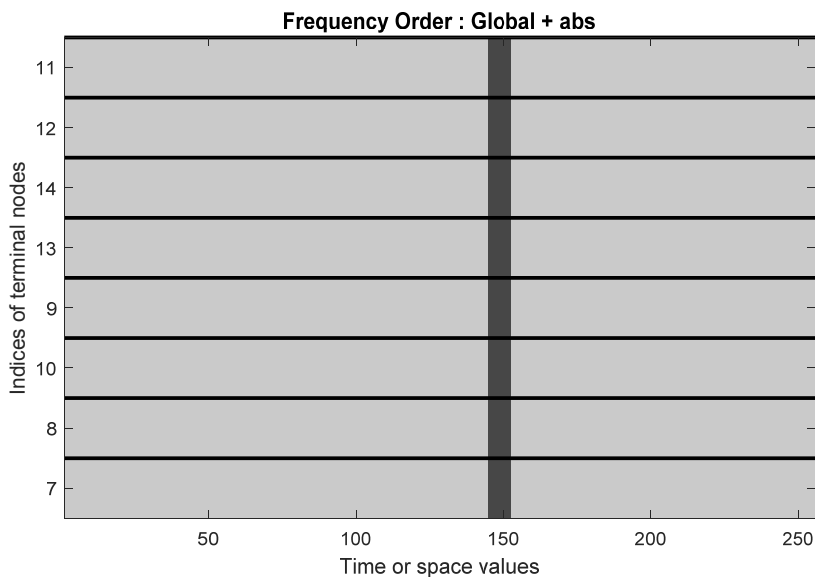


图 17-11 db1 小波分解后的时频相平面

## 17.5 变频信号的小波包分析

变频信号的表达式为  $\sin(250\pi t^2)$ ,  $0 < t < 1$ , 采样点数为 512。

【例 17-5】计算变频信号小波包变换的时频相平面。

```
%调频信号
load quachirp
x=quachirp;
figure(1);
plot(1:length(x),x);
xlabel('样本序号 n');
ylabel('幅值 A');
% 4 层 db1 小波包
wpt = wpdec(x,4,'db1');
% 小波包结构
plot(wpt);
%db1 小波基下的相平面
wpviewcf(wpt,1);
```

线性调频信号的波形如图 17-12 所示，对其进行 4 层 db1 小波的小波包分解，树结构如图 17-13 所示。

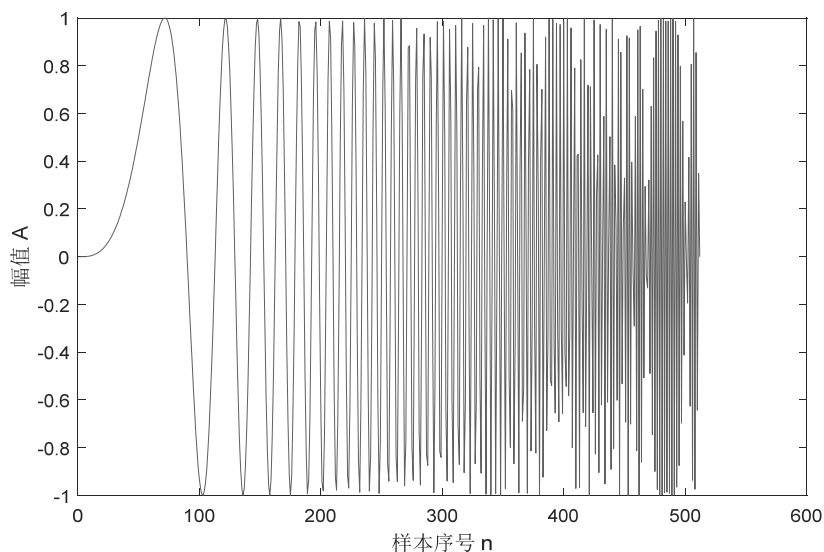


图 17-12 线性调频信号的波形

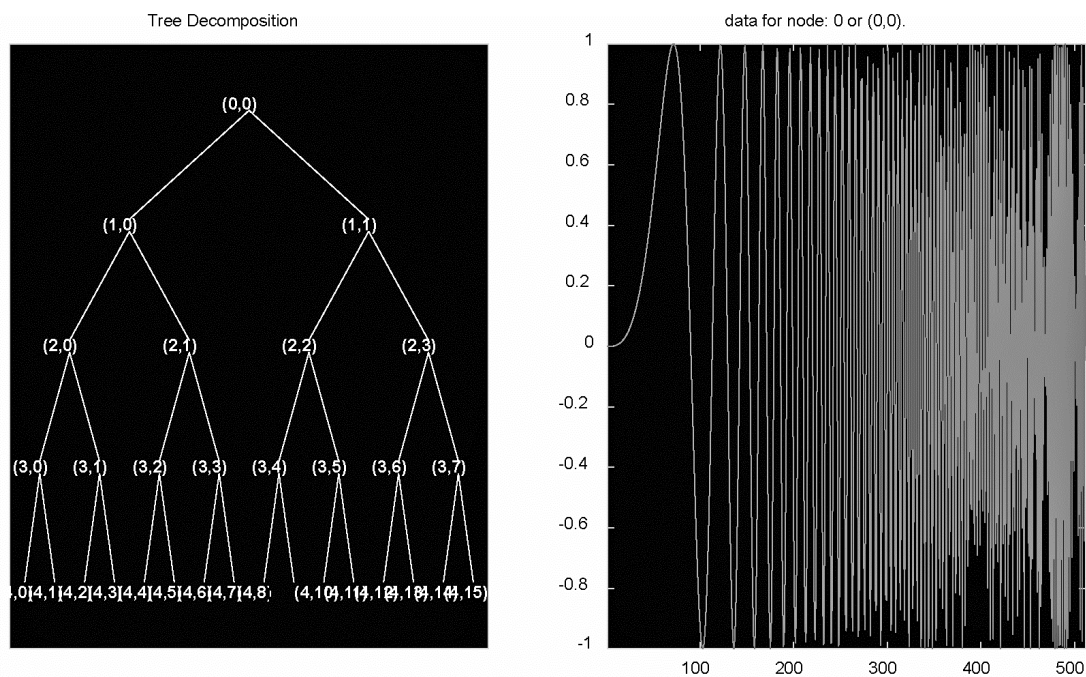


图 17-13 4 层小波包分解的树结构

db1 小波分解后的时频相平面如图 17-14 所示，可以看出其频率是随时间线性变化的，这与理论结果是一致的。

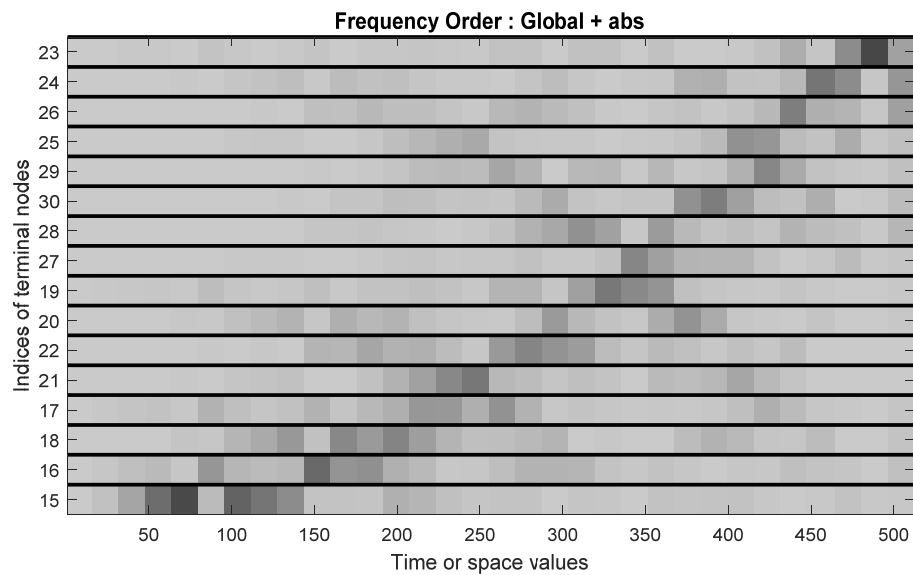


图 17-14 db1 小波分解后的时频相平面

# 第 18 章 小波在模态参数识别 与化学中的应用

## 18.1 小波在化学中的应用

小波分析在化学中的应用越来越广泛，迄今为止，有关小波变换在化学领域中应用的研究论文大约有 400 篇，其中约 75%集中在分析化学信号处理领域，包括平滑滤噪、数据压缩、基线校正、重叠信号分辨、分析化学图像处理以及与其他化学计量学方法的联合应用，另外约 25%主要集中于物理化学领域，包括量子化学、分子力学和分子动力学等。

下面举例来演示小波在化学中的应用。

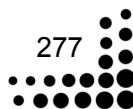
【例 18-1】假设有一高频噪声染污的色谱峰，数据如表 18-1 所示，试进行降噪处理。

表 18-1 色谱峰数据

序 列	信 号 值	序 列	信 号 值	序 列	信 号 值
1	-0.196	6	14.897	11	1.652
2	0.146	7	15.000	12	-0.234
3	0.487	8	17.524	13	0.034
4	1.778	9	9.138	14	-1.789
5	6.031	10	2.971	15	-0.128

其实现的 MATLAB 代码如下：

```
>> clear all;
x=[-0.196 0.146 0.487 1.778 6.031 14.897 15.000 17.524 9.138 2.971 1.652...
-0.234 0.034 -1.789 -0.128];
thr=thselect(x,'rigrsure'); %求阈值
y=wthresh(x,'h',thr); %作用阈值
plot(x);
grid on; hold on;
plot(y,'-o');
wname='sym6';
lev=5;
[c,l]=wavedec(x,lev,wname); %小波分解
sigma=wnoisest(c,l,1);
```





```

alpha=1.8;
thr=wbmpen(c,l,sigma,alpha);
keep=1;
xd=wdencmp('gbl',c,l,wname,lev,thr,'s',keep);    %去噪
plot(xd,'-*');
legend('原始数据','去噪方法 1','去噪方法 2');
xlabel('时间');ylabel('信号强度');

```

程序运行，结果如图 18-1 所示。

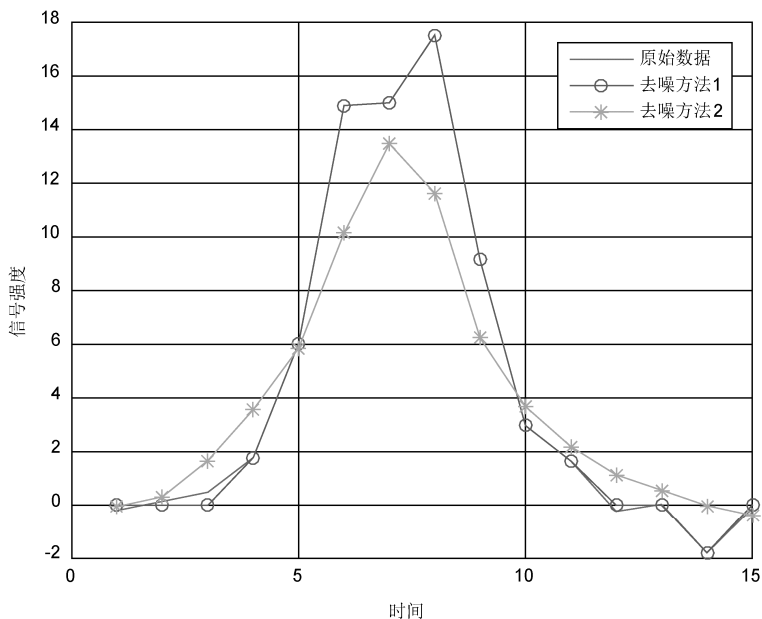


图 18-1 色谱峰数据去噪效果图

【例 18-2】如图 18-2 所示为重叠比较严重的模拟色谱图，试用小波分析进行解析。

```

>> clear all;
x=linspace(0,10,1000);
y=25*exp(-(x-5.7).^2/(2*0.5^2))+30*exp(-(x-4.3).^2/(2*0.3^2)/(2*0.3^2))+...
    20*exp(-(x-3.6).^2/(2*0.2^2))+18*exp(-(x-3.0).^2/(2*0.3^2));    %原始信号模拟
[c,l]=wavedec(y,7,'sym6');
d7=wrcoef('d',c,l,'sym6',7);
d6=wrcoef('d',c,l,'sym6',6);
d5=wrcoef('d',c,l,'sym6',5);
figure;
plot(x,y,'o');
xlabel('时间');ylabel('幅值');
axis([0 10 0 40])
figure;
subplot(4,1,1);plot(x,y);

```



```

xlabel('(a)原始信号');
subplot(4,1,2);plot(d5);
xlabel('(b)d5 信号')
subplot(4,1,3);plot(d6);
xlabel('(c)d6 信号')
subplot(4,1,4);plot(d7);
xlabel('(d)d7 信号')

```

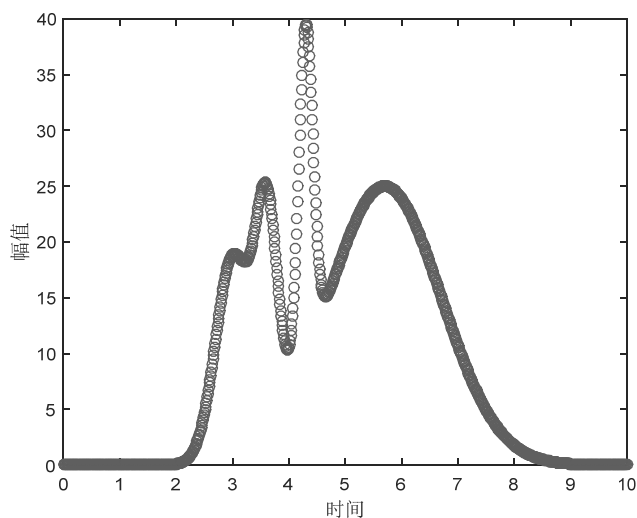


图 18-2 原始信号效果图

根据小波分析，不同尺度下的近似系数和细节系数代表着原始信号中的不同频率成分。最高频率的成分往往是噪声信号，最低频率的成分往往是基线或背景信号，而频率介于噪声和基线的成分则代表了信号的有用信息。

程序运行结果如图 18-3 所示。

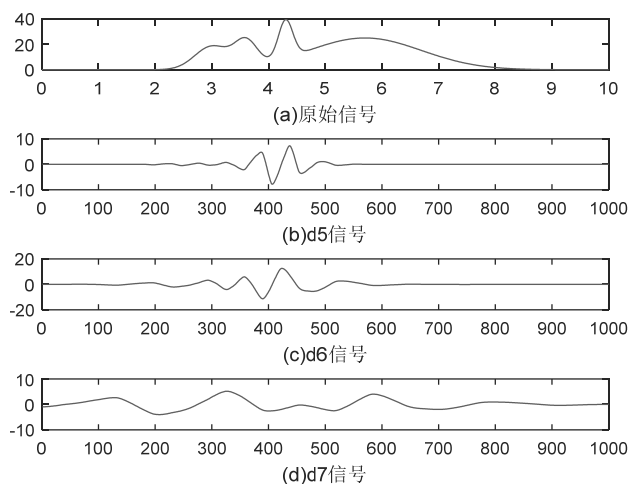
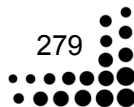


图 18-3 细节系数效果图







从图 18-3 可见,小波分解得到的  $d_6$  细节系数基本上能代表色谱图中的有用信息,可以用它做进一步的定量或定性分析。

【例 18-3】导数计算是化学信号处理的常用方法,对提高谱图的分辨率、背景信号的扣除等具有重要意义。利用小波基的性质,即不同滤波器得到的离散逼近信号之间的差别,可求得信号的导数。试对如图 18-4 所示的色谱图利用小波分析求一级导数。

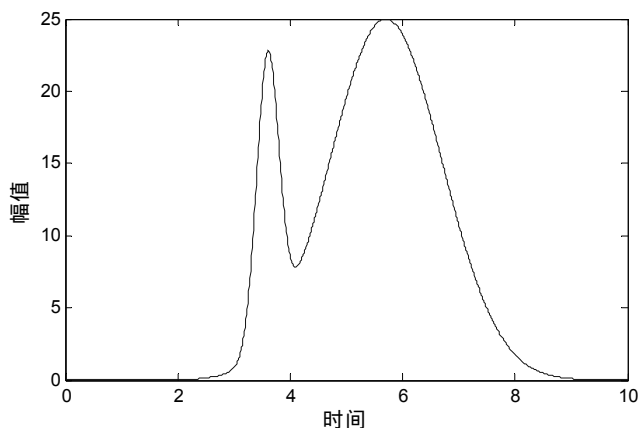


图 18-4 模拟色谱图

其实现的 MATLAB 代码如下：

```
>> clear all;
x=linspace(0,10,1000);
y=25*exp(-(x-5.7).^2/(2*0.5^2))+20*exp(-(x-3.6).^2/(2*0.2^2));    %色谱模拟
[c1,l1]=wavedec(y,7,'db4');
[c,l]=wavedec(y,7,'db2');
d25=wrcoef('d',c,l1,'db4',5);
d15=wrcoef('d',c,l,'db2',5);
figure;
plot(x,y);
xlabel('时间');ylabel('幅值');
axis([0 10 0 25])
figure;
plot(d25);hold on;
plot(d15,':');
xlabel('时间');ylabel('幅值');
legend('d25 信号','d15 信号');
```

程序运行结果如图 18-5 所示。

由图 18-5 可看出,不同长度的 db 小波基对同一信号进行分解时所得到的细节系数  $d_5$  存在着偏移,这种偏移对应于各数据点的导数。

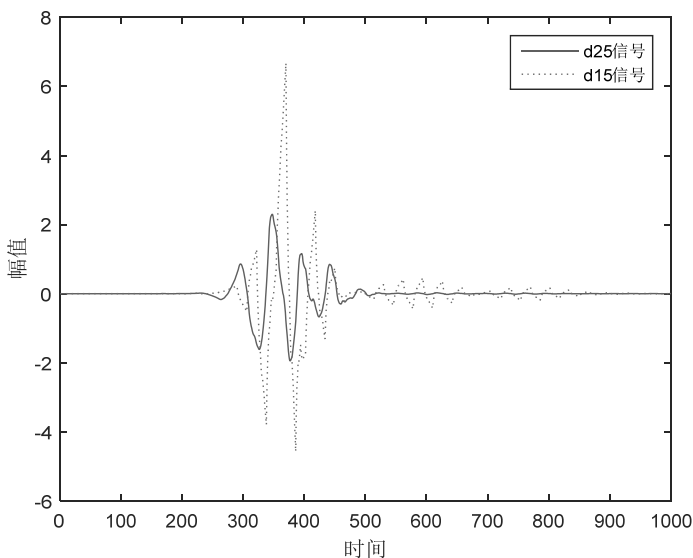


图 18-5 小波分析求一级导数

## 18.2 模态参数识别

模态参数识别方法是近年来新出现的。该方法是将信号在时频平面上展开求解特征参数，而不是单独地在时域或频域进行处理。

### 18.2.1 模态时频辨识方法

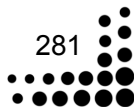
模态分析的时频方法最早出现的是 Hilbert 变换方法，由于这种方法存在较大缺陷，只存在了很短的时间就被其他时频方法所取代。时频方法现在发展较快的是利用经验模态分解方法（EMD 方法）求解模态参数的 HHT 方法，另外一种连续小波模态参数识别方法。

经验模态分析方法最早是用于非平稳信号处理，信号经 EMD 分解后各分量 IMF (Intrinsic Mode Function) 都是平稳的，可以进一步进行 Hilbert 变换得到 Hilbert 谱，由此得到的 Hilbert 谱可准确地反映出物理过程中能量在各种频率尺度及时间上的分布。

HHT 方法的本质是对一个信号进行平稳化处理分解，产生具有信号的不同特征尺度的本征模函数 IMF 分量。对于非平稳信号，对其直接进行 Hilbert 变换本身没有意义。

谱是一个三维（时间-频率-局部振幅）谱形。在线性框架下，HHT 谱与小波谱有相同的表现特性，与 FFT 谱相比，从 Hilbert 谱中不仅可以看出幅值，而且可以看出频率随时间的变化情况，这是 FFT 方法所不具有的优点。

HHT 方法能客观地处理一类非线性问题，所得到的三维谱形能准确地用于波内调制机制反映出的系统非线性变化特征，这也是其优点之一。但是，利用 HHT 方法进行模态参数识别时，在系统内阻尼比较大的情况下，会由于 Hilbert 方法本身的处理不力而出现较大的误差，而且该方法存在边缘效应问题。





系统模态参数识别的连续小波变换方法研究是国外从 1997 年开始发展起来的一个模态分析领域的新方向。该方法从本质上讲,是利用了小波函数的衰减特性,该特性与机械系统自由响应信号的衰减在形式上有相似的地方。根据平稳相位理论,对自由响应信号进行连续小波变换可以在时间-尺度(时间-频率)平面上形成一条清晰的脊线,提取该脊线上的小波变换系数,就可以分离出系统的模态参数。

### 18.2.2 小波脊线提取

在小波分析的研究中已经发现了许多合适的小波函数,如 Haar 小波、样条函数等,它们都有自己适合的应用领域。由测不准定理可知,不可能求得一个窗函数具有小于或等于 Gaussian 函数作为窗函数时窗的面积。

信号在连续小波平面上的分布会呈现类似地形图中山脊的形状,称为小波脊线。理论分析表明,沿着小波变换时频平面中脊线上分布的参数与原始信号之间有着很强的相似性,能够用来描述原始信号的重要参数。

脊线上分布数据的起伏变化、脊线的位置都有实际的物理意义,直接对应着信号的幅值与频率的变化。

时频平面上的小波脊线可以从小波变换的幅值或相位中进行提取。理论上讲,以小波变换的相位变化提取脊线更为精确,但是由于实际应用中该方法涉及相位求导,数值计算的结果可能导致结果并不十分理想。

### 18.2.3 改进 HHT 瞬时特征分析

希尔伯特-黄变换(Hilbert-Huang Transform, HHT)时频分析方法是 Norden E. Huang 等人于 1998 年提出的一种新的信号分析方法,特别适于非线性、非平稳信号的时频分析。

HHT 的主要目的是研究信号的瞬时频率,它的最大特色是通过对信号的经验模态分解(Empirical Mode Decomposition, EMD)使非平稳信号平稳化,从而使瞬时频率变得有意义,进而导出有意义的希尔伯特变换时频谱,这种时频分析方法无须采用信号的先验知识,基函数本身就是自适应地从原信号中分解而得的,具有后验、自适应性。

HHT 像其他信号分析方法一样,也存在一些不足之处。通过 EMD 得到的第一个高频 IMF 分量可能覆盖着很宽的频率范围,并不是单分量信号;在低频区域可能产生虚假 IMF 分量。尽管如此,HHT 也是在一定理论基础上提出来的,只是其理论依据尚不完备,有些方法还仅仅是依据试验提出来的。

因此,即使 HHT 已在许多工程应用实践中证明能够获得奇妙的效果,但其理论依据的进一步研究和理论框架的进一步完善是不可回避的。

### 18.2.4 模态参数识别的应用

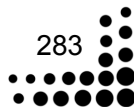
下面通过几个实例来了解怎样实现模态参数的识别。



【例 18-4】为了验证 EMD 方法处理高频数据，利用 MATLAB 对原始信号进行 EMD 分解，可实现数据的各个 IMF 分量和瞬时频率，并能对 Hilbert 时谱进行刻画。

(1) 第一步，由 5 个正弦信号合成 1 个信号：

```
>> clear all;
%由 5 个正弦信号合成一个信号
f1=450;f21=300;f22=100;f3=210;f4=250;f5=160;
Fs=1000;
for i=1:Fs+1
    t(i)=(i-1)/Fs;
    if 0.25<t(i)&t(i)<=0.5
        Ut1=1;
    else Ut1=0;
    end
    if 0.5<t(i)&t(i)<=0.75
        Ut2=1;
    else Ut2=0;
    end
    if 0.75<t(i)&t(i)<=1
        Ut3=1;
    else Ut3=0;
    end
    if 0<t(i)&t(i)<=0.25
        Ut4=1;
    else Ut4=0;
    end
    x1(i)=sin(2*pi*f1*t(i));
    x2(i)=(sin(2*pi*f21*t(i)))*Ut1;
    x3(i)=5*sin(2*pi*f3*t(i))*Ut2;
    x4(i)=sin(2*pi*f4*t(i))*Ut3;
    x5(i)=sin(2*pi*f5*t(i))*Ut4;
end
%绘制合成信号
subplot(611);plot(t,x1);
xlabel('时间');ylabel('x1');
subplot(612);plot(t,x1);
xlabel('时间');ylabel('x2');
subplot(613);plot(t,x3);
xlabel('时间');ylabel('x3');
subplot(614);plot(t,x4);
xlabel('时间');ylabel('x4');
subplot(615);plot(t,x5);
xlabel('时间');ylabel('x5');
```





```
x=x1+x2+x3+x4+x5;  
subplot(616);plot(t,x);  
xlabel('时间');ylabel('x');
```

程序运行结果如图 18-6 所示。

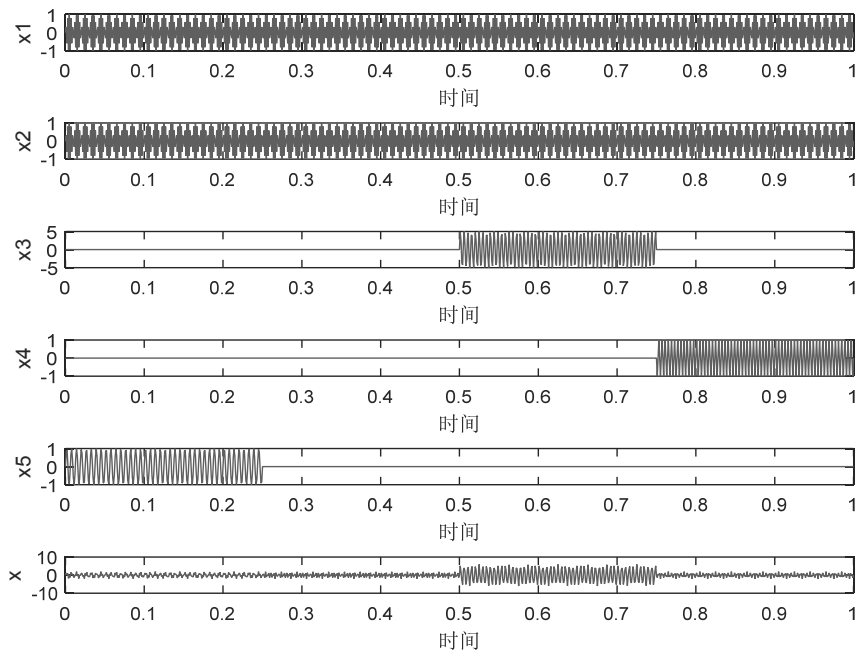


图 18-6 合成信号

(2) 第二步, 编写 HHT 函数代码:

```
function plot_hht(x,Ts)  
%x 为输入信号  
%Ts 为频率  
%调用自定义函数 emd  
imf=emd(x);  
for k=1:length(imf)  
    b(k)=sum(imf{k}.*imf{k});  
    th=angle(hilbert(imf{k}));  
    d{k}=diff(th)/Ts/(2*pi);  
end  
[u,v]=sort(-b);  
b=1-b/max(b);  
%绘制时频图  
N=length(x);  
c=linspace(0,(N-2)*Ts,N-1);  
for k=v(1:2)  
    figure;
```



```

plot(c,d{k},'r','Color',b([k k k]),'MarkerSize',3);
set(gca,'FontSize',8,'XLim',[0 c(end)],'YLim',[0 1/2/Ts]);
xlabel('时间');ylabel('频率');
end
%绘制 IMF 图
M=length(imf);
N=length(x);
c=linspace(0,(N-1)*Ts,N);
for k1=0:4:M-1
    figure;
    for k2=1:min(4,M-k1),
        subplot(4,1,k2);
        plot(c,imf{k1+k2});
        set(gca,'FontSize',8,'XLim',[0,c(end)]);
    end
    xlabel('时间');
end
end

```

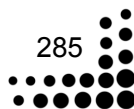
(3) 第三步, 编写经验模态函数, 代码如下:

```

function imf=emd(x)
%经验模态函数
%调用自定义的 findpeaks 函数
x=transpose(x(:));
imf=[];
while ~ ismonotonic(x) %自定义函数
    x1=x;
    sd=Inf;
    while(sd>0.1)|isimf(x1) %isimf 为自定义函数
        s1=getspline(x1); %getspline 为自定义函数
        s2=-getspline(-x1);
        x2=x1-(s1+s2)/2;
        sd=sum((x1-x2).^2)/sum(x2.^2);
        x1=x2;
    end
    imf{end+1}=x1;
    x=x-x1;
end
imf{end+1}=x;

function u=ismonotonic(x)
u1=length(findpeaks(x))*length(findpeaks(-x));
if u1>0,
    u=0;
end

```





```
else u=1;
end

function u=isimf(x)
N=length(x);
u1=sum(x(1:N-1).*x(2:N)<0);
u2=length(findpeaks(x))+length(findpeaks(-x));
if abs(u1-u2)>1
    u=0;
else u=1;
end

function s=getspline(x)
N=length(x);
p=findpeaks(x);
s=spline([0 p N+1],[0 x(p) 0],1:N);
```

(4) 第四步, 编写调用函数 findpeaks, 代码如下:

```
function n=findpeaks(x)
%找到峰值点
n=find(diff(diff(x)>0)<0);
u=find(x(n+1)>x(n));
n(u)=n(u)+1;
```

【例 18-5】利用连续小波变换提取信号的脊线。

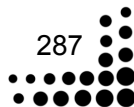
```
>> clear all;
load nearbrk;
y=nearbrk(200:240);
format long
f0=1;eps=3.0e-7;
t0=0;k=1;
N=length(y);
figure; %效果如图 18-7 所示
plot(y);
defT=1/(N-1);
a0(1:N)=1:N;
phi0=0;
ar(1:N)=ones(1,N);
t(1:N)=zeros(1,N);
phi(1:N,1:2)=zeros(N,2);
det_phi(1:N)=zeros(1,N);
real_wef(1:N,1:2)=zeros(N,2);
im_wef(1:N,1:2)=zeros(N,2);
```



```

wef(1:N,1:2)=zeros(N,2);
temp(1:N,1:2)=zeros(N,2);
fr(1:N)=zeros(1,N);
a(1:N)=zeros(1,N);
aa(1:N)=zeros(1,N);
%迭代
figure;      %效果如图 18-8 所示
for k=1:8
%连续小波变换 (复小波)
    wef(k,1:2)=cwt([y(k),y(k+1)],a0(k),'cmor4-1');
    for i=1:2
        real_wef(k,i)=real(wef(k,i));
        im_wef(k,i)=imag(wef(k,i));
        if real_wef(k,i)==0 & im_wef(k,i)==0
            temp(k,i)=0;
        else
            temp(k,i)=im_wef(k,i)/real_wef(k,i);
        end
        phi(k,i)=atan(temp(k,i));
    end
    det_phi(k)=phi(k,1)-phi(k,2);
    if det_phi(k)==0
        ar(k)=1;
    else
        ar(k)=detT*f0/det_phi(k);
    end
    eer(k)=abs((ar(k)-a0(k))/a0(k));
    a(k)=abs(f0/(2*5.14*ar(k)));
    %生成新的尺度
    if eer(k)<eps;
        a0(k)=abs(a(k));
        k=k+1;
    else a0(k)=abs(a(k));
    end
    aa=[a0 a0(N)];
    subplot(8,1,k);
    plot(aa);
end

```





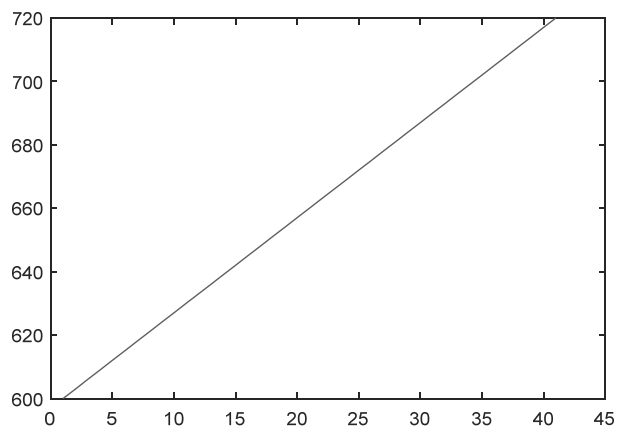


图 18-7 原始信号

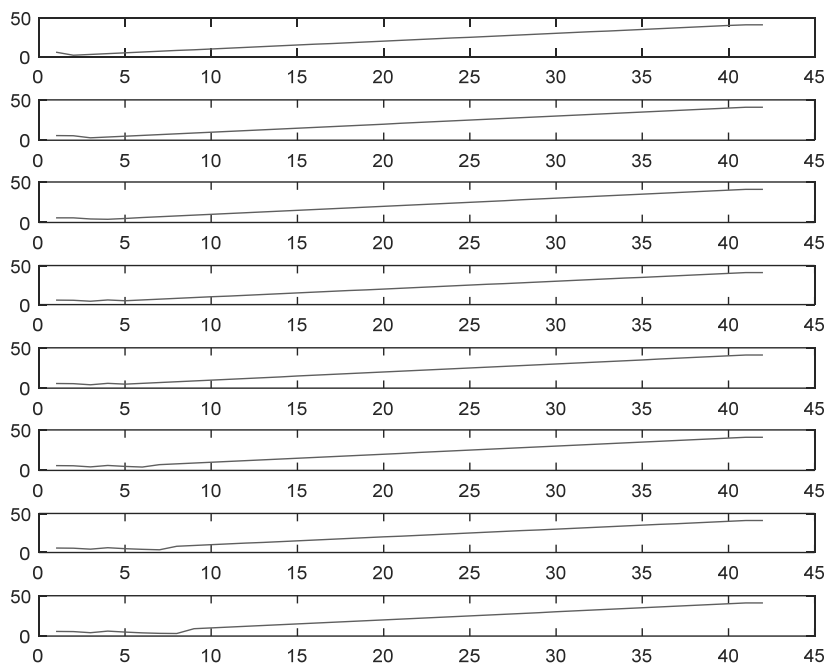


图 18-8 提取脊线

## 第 19 章 小波变换图像测试分析

### 19.1 小波变换对图像压缩的步骤

根据 Haar 函数定义,可得出当  $N=2$  时,哈尔 (Haar) 正规化变换矩阵为  $\frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$ 。

因为 Haar 矩阵是正交矩阵,具有分离变换性质,所以对二维的像素矩阵,可由连续 2 次运用一维的 Haar 小波变换来实现,叫作标准分解,如果交替地对每行和第一列像素值进行变换,则称为非标准分解。

利用矩阵形式的优点,对  $1 \times N$  的像素矩阵分解成若干个  $1 \times 2$  的矩阵与上述  $N=2$  的 Haar 正规化变换矩阵做一维 Haar 小波变换,可减少计算量,实现 Harr 小波分解。由于正规化的 Haar 变换矩阵为对称变换矩阵,其逆变换矩阵和正变换相同,因此,只要把原来每次变换后得到的矩阵数值再一次变换,就可以实现重构。

Haar 小波在时域上是不连续的,因此分析性能并不很好用,但它计算简单。后文示例中将采用非标准分解方法,在变换矩阵中,第一列变换得到图像像素均值,为图像像素低频分量,第二列得到图像像素差值,为高频分量,原像素值第  $i$  对像素分解的低频和高频分量值分别存在矩阵的  $i$  和  $N/2+i$  处。重构时取回这两个数据,再与逆变换矩阵相乘存回原处,则实现重构。

小波变换对图像压缩有以下几个主要步骤。

(1) 利用离散小波变换将图像分解为低频分量、高频的水平边缘分量、垂直边缘分量和对角边缘分量。

(2) 对低频和高频图像根据人类的视觉生理和心理特点做不同的量化和编码处理,进行压缩。

(3) 利用小波逆变换还原出原来的图像。

其中的量化工作有很多方式,这里采用阈值的设置,对采用不同小波变换后得到的低频和高频图像设置不同的阈值后得到的分解图像中含有“0”数目及重构产生的不同图像文件大小做分析,即为本次阈值测试的目的。程序用 MATLAB 中的小波函数分解图像,设置阈值后再重构保存图像,比较不同的阈值设置的测试结果。

此外,由于并不要求对分解图像做进一步的量化及编码处理后压缩存放,而是重构后存放,所以并不能对不同小波的压缩率的好坏做出结论,只能根据测试结果及小波定义做一些概括性的分析。并且, MATLAB 中保存的 TIF 格式图像文件,与其他程序保存的 TIF 文件存在偏差。这里为保证对比的一致性,对真彩色图像先用 MATLAB 对图像文件读入后保存,



再做测试，保证原始图像与重构图像存放条件的一致性。对索引图像，因为其读入的图像矩阵数值并不是图像颜色值，对其做测试后重构的图像失真程度严重，不具有实际意义，最终还是转换为真彩色图像进行比较。

## 19.2 实例说明

(1) 本例对原始图像进行 3 级非标准小波分解与重构，编写的主要处理程序有：

多级非标准 Haar 小波分解子程序 `nstdhaardec2.m`；

多级非标准 Haar 小波重构子程序 `nstdhaarrec2.m`；

多级非标准小波分解子程序 `mydwt2.m`，既可以用于 Haar 小波，也可用于 db9 小波，只需要修改对应的参数即可；

多级非标准小波重构子程序 `myidwt2.m`，既可用于 Haar 小波，也可用于 db9 小波。

(2) 此外，源程序中还给出了 Haar 小波 3 级非标准规格化分解和重构过程子程序 `nstdhaardemo.m`。

(3) 在本例程序中，需要对输入图像进行处理，输出不同格式的图像，下面对示例中使用的图像格式处理进行说明。

如果输入是彩色真彩图片，则载入的图像数值矩阵  $X$  为三维矩阵，有 R、G、B 三个分量的二维数值，在进行小波分解与重构时，需对每维数据都做分解与重构，然后转换为 `uint8` 数据类型，并显示其彩色图像。如果输入的是黑白图片，即输入的数据矩阵  $X$  为二维矩阵，则只对其做分解与重构即可。

(4) 使用小波变换在不同阈值下压缩图像的测试程序为 `thresholdtestdemo.m`。

(5) 对不同小波在不同分解模式下进行 3 级分解与重构图像的子程序是 `modetest.m`。

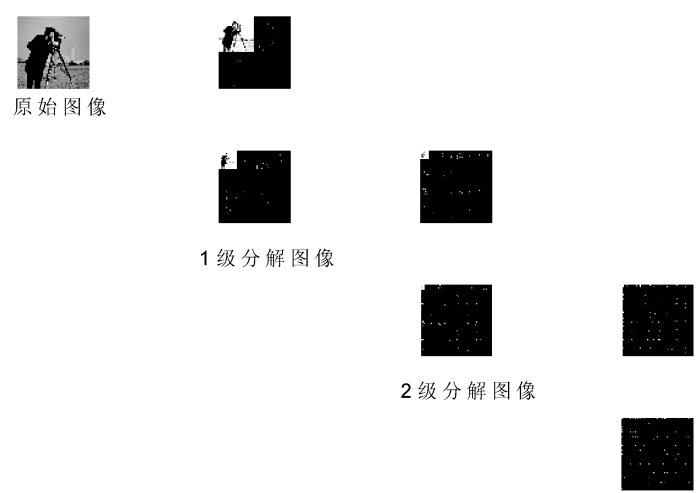
## 19.3 输出结果与分析

在本例中，应用 Haar 小波对原始图像 `cameraman.tif` 进行 3 级非标准规格化分解和重构，其图像分解过程如图 19-1 (a) 所示，而图像重构过程如图 19-1 (b) 所示。

此外，本例中还应用小波变换子程序 `thresholdtestdemo.m` 在不同阈值下对图像进行压缩测试。图 19-2 和图 19-3 分别说明了应用 Haar 小波、db9 小波分别在阈值为 0、5、10、20 情况下的 1 级分解测试结果，同时也分别给出了在阈值为 0、20、40、80 情况下的 3 级分解测试结果。

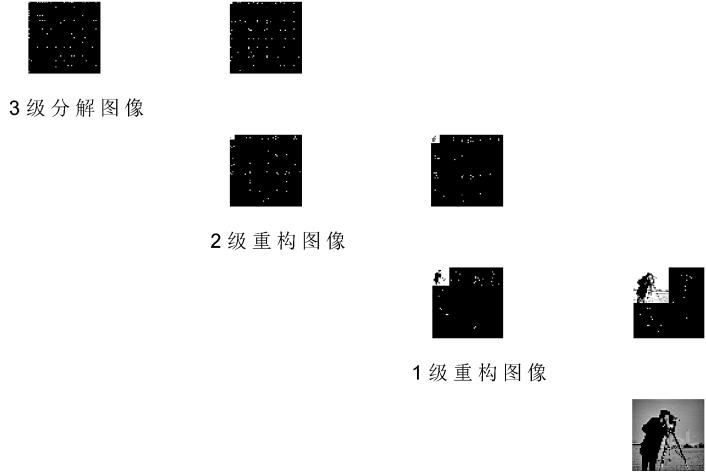
从测试总结：测试数据可见，阈值取得越大，分解设置阈值后的矩阵系数为“0”的数目就越多。分解级数越多，同一阈值下系数为“0”的数目也越多。但同时阈值取得越大，图像重构后的失真程度也越厉害。例如，测试例子中，3 级分解后重构，在  $T=40$  时可以看出失真情况，在  $T=80$  的重构图像上明显看到图像的失真情况比较严重。

Haar小波3级非标准规格化分解过程



(a) Haar小波3级非标准规格化图像分解过程 (JPG)

Haar小波3级非标准规格化分解重构过程



(b) Haar小波3级非标准规格化图像重构过程 (JPG)

图 19-1 Haar小波3级非标准规格化图像分解过程与重构过程 (JPG)



Haar小波变换阈值测试 (1级)

阈值T=0 (Size:37KB)

阈值T=5 (Size:28KB)



原始图像 (Size:64KB)

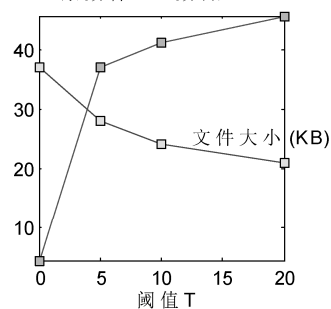
系数含“0”数目=4414

系数含“0”数目=36929

阈值T=10 (Size:24KB)

阈值T=20 (Size:21KB)

系数含“0”数目/1000



系数含“0”数目=41308

系数含“0”数目=45706

(a) 阈值 T=0、5、10、20 的测试结果(1 级分解)

Haar小波变换阈值测试 (3级)

阈值T=0 (Size:37KB)

阈值T=20 (Size:16KB)



原始图像 (Size:64KB)

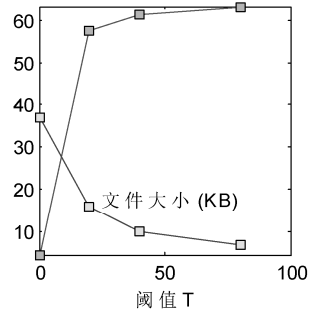
系数含“0”数目=4727

系数含“0”数目=57452

阈值T=40 (Size:10KB)

阈值T=80 (Size:7KB)

系数含“0”数目/1000



系数含“0”数目=61174

系数含“0”数目=63144

(b) 阈值 T=0、20、40、80 的测试结果(3 级分解)

图 19-2 Haar 小波变换测试结果

db9 小波变换 阈值测试 (1 级)



原始图像 (Size: 64KB)  
阈值 T=10 (Size: 64KB)



系数含“0”数目=46568

阈值 T=5 (Size: 37KB)



系数含“0”数目=0  
阈值 T=20 (Size: 35KB)

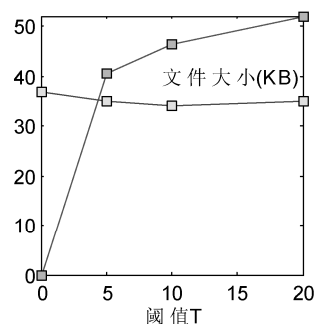


系数含“0”数目=52148

阈值 T=5 (Size: 35KB)



系数含“0”数目=40613  
系数含“0”数目/1000



(a) 阈值 T=0、5、10、20 的测试结果 (1 级分解)

db9 小波变换 阈值测试 (3 级)



原始图像 (Size: 64KB)  
阈值 T=40 (Size: 31KB)



系数含“0”数目=75485

阈值 T=0 (Size: 37KB)



系数含“0”数目=0  
阈值 T=80 (Size: 29KB)

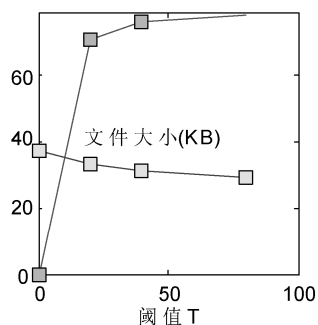


系数含“0”数目=78040

阈值 T=20 (Size: 33KB)



系数含“0”数目=70050  
系数含“0”数目/1000



(b) 阈值 T=0、20、40、80 的测试结果 (3 级分解)

图 19-3 db9 小波变换测试结果



db9 与 Haar 小波相比较, 阈值取得越大, Haar 小波变换后设置阈值重构存放的文件就越小; 而 db9 小波, 阈值的取值对重构后存放文件大小的影响不如 Haar 小波变换明显, 而且有比原文件还大的项 ( $T=50$ )。用其他图片试验, 结果依然如此, 但阈值取得很大以后, 文件是减小的, 不过这时图像已明显失真了。由此可以得出, 阈值的设置对用 db9 小波分解重构后用 PNG 格式压缩存放, 不具有线性意义。据分析, 这与 Daubechies 滤波器本身的性质有关, 分解与重构时构造的正规性滤波器是不对称的, 不具有线性相位 (正规性条件越好, 滤波器越长), 重构过程为高低频滤波器与各个矩阵卷积后再相加来恢复图像数据的, 本身“0”阈值重构时得到的就只是与原图像数据相似的数据 (四舍五入取整才相等, 这里用四舍五入处理后保存, 不过与原来用 `uint8()` 直接转换保存结果类似, 最多差 1KB), 阈值设置后重构改变了原文件本来具有的相邻相似性数据的数量, 对采用基于词典编码思想的 (LZ77 压缩算法) PNG 格式图像压缩存放, 是明显不利的, 所以会有比原文件还大的情况出现, 但具体要看分解图像本身数据相互之间的相似程度, 文件变大的情况不是在相同阈值的条件下出现的。Haar 小波滤波器是规格化正交基, 长度较短, 对称, 滤波“0”阈值时能完整重构原图像, 即使不做四舍五入, 其误差也很小 (由上述参考最大误差值表格可以看出), 可以忽略不计, 所以即使加设阈值后, 这一现象也可能不是很明显甚至不会出现。但如果是矩阵相邻相似性数据非常多的情况, 也会有这种现象出现。随着阈值的加大, 把分解矩阵中的高频分量细节几乎全过滤掉了, 重构得到的图像类似于进行图像平滑处理, 即让与周围像素值的差比较大的像素改变为与周围像素值接近的值, 就反而对 PNG 格式保存有利了, 文件就明显比原文件小了。

从图像质量上比较, db9 小波比 Haar 小波的分析性能要好, 分解细致, 即使在失真的情况下, 也不会出现 Haar 小波那样的马赛克状, 只是模糊羽化状态。

另外, 本实例也利用子程序 `modetest.m` 对图像进行不同模式下的 3 级 Haar 小波和 db9 小波分解与重构操作, 应用的模式有 `sym` 模式和 `per` 模式, 如图 19-4 和图 19-5 所示分别是利用 Haar 小波和 db9 小波在这两种模式下进行 3 级小波分解和重构的结果。

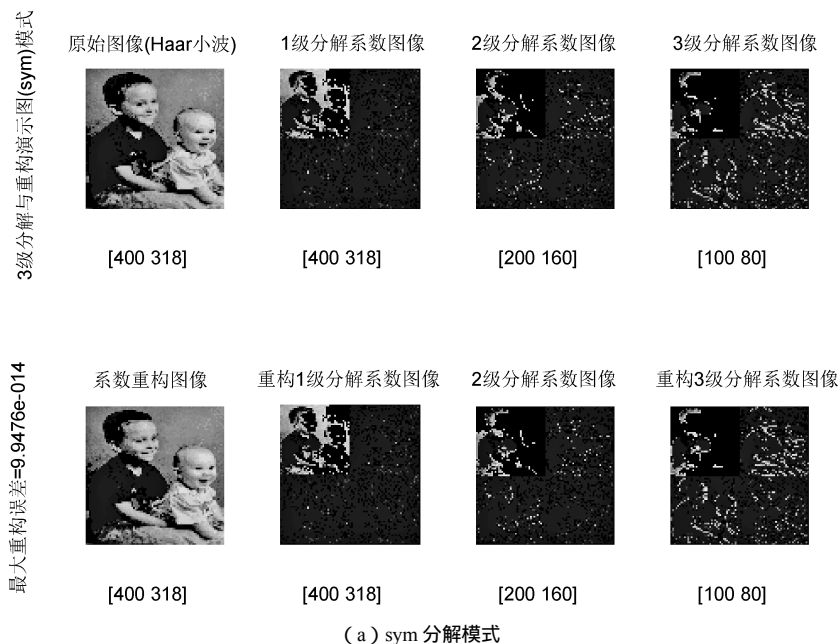


图 19-4 Haar 小波 3 级分解与重构图像

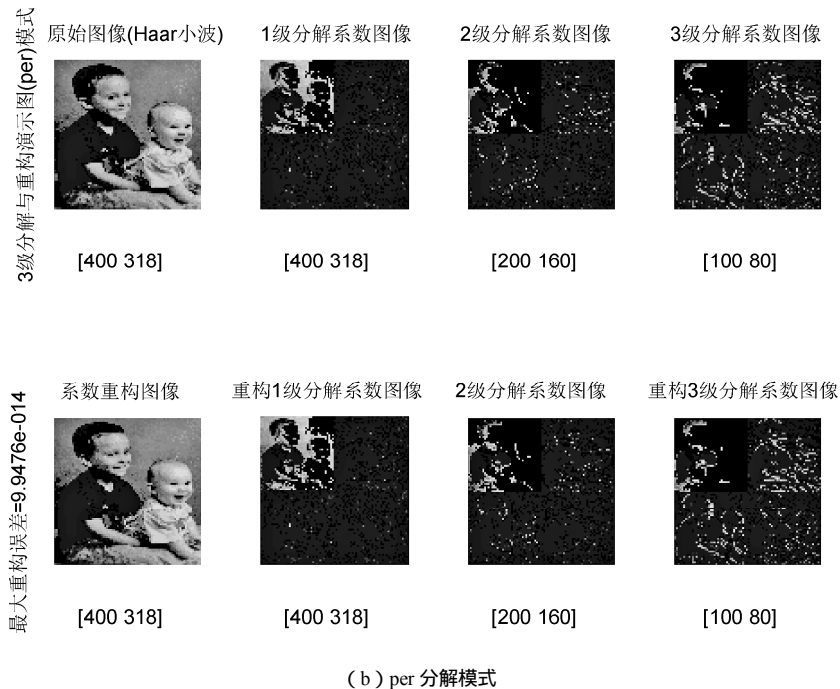


图 19-4 Haar 小波 3 级分解与重构图像 (续)



图 19-5 db9 小波 3 级分解与重构图像





图 19-5 db9 小波 3 级分解与重构图像 (续)

## 19.4 源程序

### 1. nstdhaardemo.m

```
function nstdhaardemo(imgname)
% nstdhaardemo Haar 小波 3 级非标准规格化分解与重构演示程序
% nstdhaardemo(imgname)
% 本程序作用:载入图像文件,显示图像 3 级非标准 Haar 小波规格化分解与重构过程
% 输入: imgname——要装载的图像名称(真彩色、灰度图、索引图)
% 默认图像为 color256.png

if nargin==0
    imgname='color256.png';
end
% 读入的 X 中含有被装载的图像信号,map 中含有被装载的 color
[X,map] = imread(imgname);
if ndims(X)==3
    imgcolor=1;
else
    imgcolor=0;
end
X=double(X);
```



```

h=size(X,1);

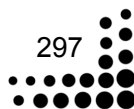
%画出原始图像
figure(1);
subplot(4,4,1);imshow(uint8(X),map);
title(' Haar 小波 3 级非标准规格化分解过程');
xlabel('原始图像');
axis square

position=2;
A=X;
%依次序画出 3 级非标准规格化分解过程中对应的行变换、列变换图像
for i=1:3
A011=decompose(A,imgcolor,1,1,h);
subplot(4,4,position);imshow(uint8(A011),map);axis square
A012=decompose(A011,imgcolor,1,2,h);
subplot(4,4,position+4);imshow(uint8(A012),map);axis square
xlabel(strcat(int2str(i),'级分解图像'));
A=A012;
h=h/2;
position=position+5;
end
xlabel('Haar 小波 3 级非标准规格化分解图像');

%画出 3 级分解图像
figure(2);
subplot(4,4,1);imshow(uint8(A),map);
title('Haar 小波 3 级非标准规格化分解重构过程');
xlabel('3 级分解图像');
axis square
position=2;

%依次序画出 3 级非标准规格化分解后重构过程中对应的行变换、列变换图像
if imgcolor
h=size(A(:,1),1)/4;
else
h=size(A,1)/4;
end
for i=1:3
RX11=reconstruct(A,imgcolor,1,1,h);
subplot(4,4,position);imshow(uint8(RX11),map);axis square
RX12=reconstruct(RX11,imgcolor,1,2,h);
subplot(4,4,position+4);imshow(uint8(RX12),map);axis square

```





```
xlabel(strcat(int2str(3-i),'级重构图像'));
A=RX12;
h=h*2;
position=position+5;
end
xlabel('Haar 小波 3 级非标准分解后重构图像');
figure(1);

%-----
% 内部程序
%-----
function C=decompose(A,imgcolor,level,roworcol,h)

if imgcolor
Ar=A(:,:,1);
Ag=A(:,:,2);
Ab=A(:,:,3);
Cr=nstdhaardec2(Ar,level,roworcol,h);
Cg=nstdhaardec2(Ag,level,roworcol,h);
Cb=nstdhaardec2(Ab,level,roworcol,h);
C(:,:,1)=Cr;
C(:,:,2)=Cg;
C(:,:,3)=Cb;
else
C=nstdhaardec2(A,level,roworcol,h);
end

%-----
% 内部程序
%-----
function C=reconstruct(A,imgcolor,level,roworcol,h)

if imgcolor
Ar=A(:,:,1);
Ag=A(:,:,2);
Ab=A(:,:,3);
Cr=nstdhaarrec2(Ar,level,roworcol,h);
Cg=nstdhaarrec2(Ag,level,roworcol,h);
Cb=nstdhaarrec2(Ab,level,roworcol,h);
C(:,:,1)=Cr;
C(:,:,2)=Cg;
C(:,:,3)=Cb;
else
```



```
C=nstdhaarrec2(A,level,roworcol,h);
End
```

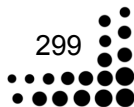
## 2. thresholdtestdemo.m

```
function thresholdtestdemo(imgname,wavename,level,deta)
% thresholdtestdemo 小波变换在不同阈值下重构的测试演示程序
% thresholdtestdemo(imgname,wavename,level,deta)
% 本程序作用:载入图像信号,使用小波变换,比较显示不同阈值下的重构图像及相关信息
% 输入: imgname——要装载的图像名称
% wavename——变换小波名称
% level——小波分解级数
% deta——测试阈值增加值
% 默认时:变换图像为 color256.png, 3 级 Haar 小波变换 deta=5

if nargin==0
imgname='color256.png';
wavename='haar';
level=1;
deta=90;
else
if nargin==1
wavename='haar';
level=3;
deta=5;
else
if nargin==2
level=3;
deta=5;
else
if nargin==3
deta=5;
end
end
end
end

% 读入的 X 中含有被装载的图像信号,map 中含有被装载的 color
[X,map] = imread(imgname);

% 检测图像格式
% 输入矩阵 X 为三维时为真彩色图像。RGB 三个分量需分别分解与重构
% 输入图像为索引图像时(map 不为空), 转换为 RGB 真彩色图像, 再做分解与重构, 因为索引图
    每个像素的颜色值
```





% 不是矩阵的数值,真正应该显示的颜色为读入的矩阵 X 上的值通过颜色表 map 变换得到的,如果直接用来做变换,即有可能会发生失真情况

```
if isempty(map)
emp=1;
else
X=ind2rgb(X,map)*255;
emp=0;
end
if ndims(X)==3
imgcolor=1;
Xr=X(:,:,1);
Xg=X(:,:,2);
Xb=X(:,:,3);
else
imgcolor=0;
end

info=imfinfo(imgname);
ss=round((info(1).FileSize)/1024);
sname=size(imgname);
orgname=imgname(1:sname(2)-4);
nfile=struct('name','');
X=double(X);
% 画出原始图像
figure(1);
subplot(2,3,1);
imshow(uint8(X));colormap(map);
title(strcat(wavename,'小波变换阈值测试(',mat2str(level),'级'));
xlabel(strcat('原始图像(Size: ',mat2str(ss),'KB')));
axis square
% 小波分解
if imgcolor
[Cr,Sr]=wavedec2(Xr,level,wavename);
[Cg,Sg]=wavedec2(Xg,level,wavename);
[Cb,Sb]=wavedec2(Xb,level,wavename);
C(1,:)=Cr;
C(2,:)=Cg;
C(3,:)=Cb;
else
[C,S]=wavedec2(X,level,wavename);
end
```



```
% 在不同阈值下测试
T(1)=0;
for i=1:4
% 用 0 置换矩阵中绝对值小于阈值的数值
if T(i)>0
C(find(abs(C)<=T(i)))=0;
%C=wthresh(C,'h',T(i));
end

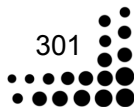
% 计算 “ 0 ” 的个数
zeronum(i)=prod(size(find(C==0)));

% 重构不同阈值下图像
if imgcolor
Cr=C(1,:);
Cg=C(2,:);
Cb=C(3,:);
Ar=waverec2(Cr,Sr,wavename);
Ag=waverec2(Cg,Sg,wavename);
Ab=waverec2(Cb,Sb,wavename);
A(:,:,1)=uint8(round(Ar));
A(:,:,2)=uint8(round(Ag));
A(:,:,3)=uint8(round(Ab));
else
A=waverec2(C,S,wavename);
A=uint8(round(A));
end

% 存储新图像,图像名为原图像名+小波名+阈值大小
nfile(i).name=strcat(orgname,'_',wavename,'_',mat2str(T(i)),'.png');
imwrite(A,nfile(i).name,'png');

% 获取图像大小信息
info=imfinfo(nfile(i).name);
if info(1).FileSize>1024
sfile(i)=round((info(1).FileSize)/1024);
sunit='KB';
else
sfile(i)=info(1).FileSize;
sunit='B';
end

% 画出重构图像
```





```
figure(1);
subplot(2,3,i+1);
imshow(uint8(A));
title(strcat('阈值 T=',int2str(T(i)), ' (Size:',int2str(sfile(i)),sunit,')'));
xlabel(strcat('系数含"0"数目=',int2str(zeronum(i))));
axis square
if i<3
    T(i+1)=T(i)+deta;
else
    if i==3
        T(i+1)=T(i)+2*deta;
    end
end;
end

% 画出不同阈值对应的文件大小及系数为“0”的关系图
if and(or(min(zeronum)>1000,zeronum(1)==0),zeronum(4)/1000>sfile(4))
    zeronum=zeronum/1000;
    overth=1;
else
    overth=0;
end;
subplot(2,3,6);
plot(T,zeronum,'-rs','LineWidth',1,...
'MarkerEdgeColor','k',...
'MarkerFaceColor','g',...
'MarkerSize',5);
hold on;
plot(T,sfile,'-bs','LineWidth',1,...
'MarkerEdgeColor','k',...
'MarkerFaceColor','y',...
'MarkerSize',5);
xlabel(' 阈值 T ');
axis([0 100 min(sfile(1),zeronum(1)) max(sfile(4),zeronum(4))]);

meant=(zeronum(4)-sfile(4))/8;
if overth
    text(10,zeronum(4)+meant,'系数含"0"数目/1000');
else
    text(10,zeronum(4)+meant,'系数含"0"数目');
end
text(15,sfile(4)+meant,strcat('文件大小\(',sunit,')'));
axis square
```

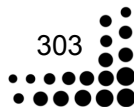


## 3. modetest.m

```

function modetest()
% 不同小波在不同的分解模式下 3 级分解与重构图像
% 图像为 kids.tif, Haar 小波, per 分解模式
% 装载图像为黑白索引图像, X 中含有被装载的信号, map 中含有被装载的 color
imgname='kids.tif';           % 输入: imgname 图像文件
wavename='haar';              % wavename 小波名称
mode='per';
[X,map]=imread(imgname);
deccof=struct('cA',[],'cH',[],'cV',[],'cD',[]);
reccof=struct('RX',[]);
sX=size(X);
nbcot=size(map,1);
X=double(X);
% 画出原始图像
figure(1);
subplot(241);imshow(uint8(X));colormap(map);
title(strcat('原始图像(',wavename,'小波)'),FontSize,7.5);
ylabel(strcat('3 级分解与重构演示图(',mode,'模式)'),FontSize,7.5);
xlabel(mat2str(sX));
axis square
DX=X;
deccof(1).cA=X;
for i=2:4
% 用小波函数进行分解
[deccof(i).cA,deccof(i).cH,deccof(i).cV,deccof(i).cD]=dwt2(DX,wavename,'mode',mode);
% 画出各分解系数对应的图像
subplot(2,4,i);
imshow([deccof(i).cA/255,deccof(i).cH/255;deccof(i).cV/255,deccof(i).cD/255;]);
colormap(map);
title(strcat(int2str(i-1),'级分解系数图像'),FontSize,7.5);
xlabel(mat2str(2*size(deccof(i).cA)));
axis square;
DX=deccof(i).cA;
end
reccof(i).RX=deccof(i).cA;
i=i+4;
for j=4:-1:2
% 画出每级重构的图像
figure(1);
subplot(2,4,i);
imshow([reccof(j).RX/255,deccof(j).cH/255;deccof(j).cV/255,deccof(j).cD/255]);

```







```
if j==3
    title(strcat(int2str(j-1),'级分解系数图像'),'FontSize',7.5);
else
    title(strcat('重构',int2str(j-1),'级分解系数图像'),'FontSize',7.5);
end
axis square;
xlabel(mat2str(2*size(reccof(j).RX)));
% 利用分解系数进行直接重构
reccof(j-1).RX=idwt2(reccof(j).RX,deccof(j).cH,deccof(j).cV,deccof(j).cD,wavename,size
    (deccof(j-1).cA),'mode','mode');
i=i-1;
end
% 检查重构精度
A0max1=max(max(abs(X-reccof(1).RX)));
A0max2=prod(size(find(abs((X-(reccof(1).RX))) ~ =0)));
subplot(245);imshow(reccof(1).RX/255);colormap(map);
title('系数重构图像','FontSize',7.5);
axis square;
xlabel(mat2str(size(reccof(1).RX)));
ylabel(strcat('最大重构误差=',num2str(A0max1)), 'FontSize',7.5);
```

#### 4. nstdhaardec2.m

```
function [a,lt]=nstdhaardec2(x,level,rorc,h)
% 二维 Haar 小波非标准规格化分解程序（多级分解）
% 作用：使用 Haar 小波对每行和每列像素值都进行小波变换
% 输入：x  载入的二维图像像素值
a=double(x);           % 输出：a  分解后的数值矩阵，大小与 x 相同
t=1;                   % 记录实际分解次数
sX=size(x);
level=1;               % 小波分解次（级）数设定值（如果设定值超过最高可分解次
                        % 数，按最高分解次数分解）
h=sX(2);               % 分解的矩阵块大小，默认为整个 x 矩阵的变换
rorc=0;                % 做行变换（1）或列变换（2），默认值为“0”
lt=level;
while and(h>1,t<=level)
    if rorc==1;
        for row=1:h
            a(row,:)=haardec(a(row,:),h);
        end
    else
        if rorc==2
            for col=1:h
                temp=haardec(a(:,col)',h);
```



```

        a(:,col)=temp';
    end
else
    for row=1:h
        a(row,:)=haardec(a(row,:),h);
    end
    for col=1:h
        temp=haardec(a(:,col)',h);
        a(:,col)=temp';
    end
end
end
h=h/2;
t=t+1;
end
if and(h<=1,lt ~ =t-1)
    lt=t-1;
end
%-----
function y=haardec(c,h)
% haardec 1-D haar decompose program
% y=haardec(c,l)
y=c;
sqrt2=sqrt(2);
h=h/2;
for i=1:h
    y(i)=(c(2*i-1)+c(2*i))/sqrt2;
    y(h+i)=(c(2*i-1)-c(2*i))/sqrt2;
end
end

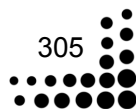
```

## 5. nstdhaarrec2.m

```

function a=nstdhaarrec2(x,level,rorc,h)
% nstdhaarrec2 二维非标准 Haar 小波规格化分解后图像重构程序（多级）
% 输出:x 载入的二维图像像素值
a=double(x); % 输出:a 重构后生成的图像像素数值矩阵，大小与 x 相同
level=1; % 分解重构层数
rorc=0; % 做行变换（1）或列变换（2），默认值为“0”，行列变换都做
h=size(x,2); % 重构的矩阵块大小，默认为整个 x 矩阵的变换
h1=h;
h2=h*(2^(level-1));
while h1<=h2
    if rorc==1;
        for j=1:h1

```





```
        tempcol=a(1:h1,j)';
        a(1:h1,j)=haarrec(tempcol,h1)';
    end
else
    if rore==2
        for i=1:h1
            temprow=a(i,1:h1);
            a(i,1:h1)=haarrec(temprow,h1);
        end
    else
        for i=1:h1
            temprow=a(i,1:h1);
            a(i,1:h1)=haarrec(temprow,h1);
        end
        for j=1:h1
            tempcol=a(1:h1,j)';
            a(1:h1,j)=haarrec(tempcol,h1)';
        end
    end
end
h1=h1*2;
end
%-----
function y=haarrec(x,h)
% haarrec 1-D haar reconstruct program
c=x;
h1=h/2;
for i=1:h1
    y(2*i-1)=(c(i)+c(h1+i))/sqrt(2);
    y(2*i)=(c(i)-c(h1+i))/sqrt(2);
end
```

## 6. mydwt2.m

```
function deccoef=mydwt2(X,wavename,N,mode)
% 2-D 多级非标准小波分解程序
% 输入:X 要分解的二维信号
% wavename 用来做分解的小波名称与 MATLAB 的 wavename 定义一致
% N 分解级数
% 说明: 数组标号对应分解级数
sX=size(X);
DX=X;
% 设置默认分解模式
mode='sym';
```



```

if sX(1)==1
    error=sprintf('%s','出错信息：分解信号需要二维矩阵')
else
    % 用小波进行分解
    for i=1:N
        [deccoeff(i).cA,deccoeff(i).cHdeccoeff(i).cdeccoeff(i).cV,deccoeff(i).cD]=dwt2(DX,wavename,
            'mode',mode);
    % cA,cH,cV,cD 分别保存低频、水平高频、垂直高频、斜线高频分解系数值
        deccoeff(i).ex_size=size(DX);
    % 输出 deccoeff 3 级分解的各级分解系数
    % ( 1x3 struct array with fields:cA,cH,cV,cD,ex_size)
        DX=deccoeff(i).cA;
    end
end
end

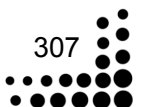
```

## 7. myidwt2.m

```

function X=myidwt2(mode)
% myidwt2 2-D 多级非级标准小波重构程序
mode='sym';
reccoeff(N+1).RX=deccoeff(N).cA;
for j=N:-1:1
    % 利用小波分解系数重构
    reccoeff(j).RX=idwt2(reccoeff(j+1).RX,deccoeff(j).cH,deccoeff(j).cV,deccoeff(j).cD,
        deccoeff(j).cD,wavename,deccoeff(j).ex_size,'mode',mode);
end
X=reccoeff(1).RX;

```



## 第 20 章 小波包分解与重构算法的应用

小波包变换是基于小波变换的进一步发展，能够提供比小波变换更高的分辨率。本章介绍小波包分析方法，从小波包的构造、去噪阈值的确定等方面进行阐述。小波包分解与小波分解相比，是一种更精细的分解方法，它不仅对图像的低频部分进行分解，也对图像的高频部分进行分解。小波包对图像分解做多分辨率分解是在小波函数对图像的分解基础上发展起来的，通过水平和垂直滤波，小波包变换将原始图像分为 4 个子带：水平和垂直方向上的低频子带、水平和垂直方向上的高频子带，继续对图像的低频子带和高频子带进行分解就可以得到图像的小波包分解树结构。

### 20.1 小波包基本理论

短时傅里叶变换对信号的频带划分是线性等间隔的。多分辨分析可以对信号进行有效的时频分解，但由于其尺度是按二进制变化的，所以在高频频段其频率分辨率较差，而在低频频段其时间分辨率较差，即对信号的频带进行指数等间隔划分（具有等  $Q$  结构）。小波包分析能够为信号提供一种更精细的分析方法，它将频带进行多层次划分，对多分辨率分析没有细分的高频部分进一步分解，并能够根据被分析信号的特征，自适应地选择相应频带，使之与信号频谱相匹配，从而提高了时-频分辨率，因此小波包具有更广泛的应用价值。

关于小波包分析的理解，我们这里以一个 3 层的分解进行说明，其小波包分解树如图 20-1 所示。

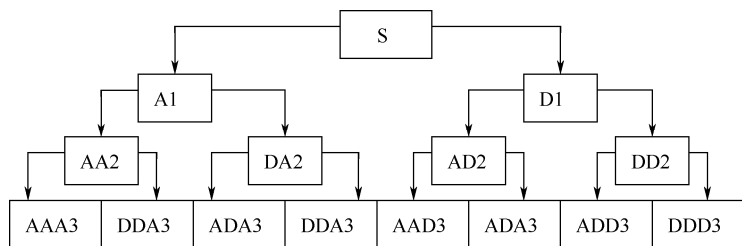


图 20-1 3 层小波包分解树

图 20-1 中，A 表示低频，D 表示高频，数字表示小波分解的层数（尺度数）。由图 20-1 可见，分解级数越大，也就是选择的小波包尺度越大，小波包系数对应的空间分辨率就越低，利用这点，可以在不同的空间分辨率上进行分析，实现图像的去噪、压缩、编码等各种处理



工作。

分解具有关系  $S=AAA3+DAA3+ADA3+DDA3+AAD3+DAD3+ADD3+DDD3$ 。

### 20.1.1 小波包理论分析

在多分辨分析中,  $L^2(R) = \bigoplus_{j \in Z} W_j$ , 表明多分辨分析是按照不同的尺度因子  $j$  把 Hilbert 空间  $L^2(R)$  分解为所有子空间  $W_j (j \in Z)$  的正交和, 其中,  $W_j$  为小波函数  $\psi(t)$  的闭包 (小波子空间)。现在, 我们希望进一步对小波子空间  $W_j$  按照二进制分式进行频率的细分, 以达到提高频率分辨率的目的。

一种自然的做法是将尺度空间  $V_j$  和小波子空间  $W_j$  用一个新的子空间  $U_j^n$  统一起来表征, 若令

$$\begin{cases} U_j^0 = V_j \\ U_j^1 = W_j \end{cases}, j \in Z$$

则 Hilbert 空间的正交分解  $V_{j+1} = V_j \oplus W_j$  即可用  $U_j^n$  的分解统一为

$$U_{j+1}^0 = U_j^0 \oplus U_j^1, j \in Z \quad (20-1)$$

定义子空间  $U_j^n$  是函数  $U_n(t)$  的闭包空间, 而  $U_j^{2n}$  是函数  $u_{2n}(t)$  的闭包空间, 并令  $u_n(t)$  满足下面的双尺度方程:

$$\begin{cases} u_{2n}(t) = \sqrt{2} \sum_{k \in Z} h(k) u_n(2t - k) \\ u_{2n+1}(t) = \sqrt{2} \sum_{k \in Z} g(k) u_n(2t - k) \end{cases} \quad (20-2)$$

式中,  $g(k) = (-1)^k h(1-k)$ , 即两系数也具有正交关系。当  $n=0$  时, 式 (20-2) 可写为

$$\begin{cases} u_0(t) = \sum_{k \in Z} h_k u_0(2t - k) \\ u_1(t) = \sum_{k \in Z} g_k u_0(2t - k) \end{cases} \quad (20-3)$$

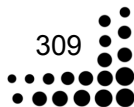
在多分辨分析中,  $\phi(t)$  和  $\psi(t)$  满足双尺度方程

$$\begin{cases} \phi(t) = \sum_{k \in Z} h_k \phi(2t - k), \{h_k\}_{k \in Z} \in l^2 \\ \psi(t) = \sum_{k \in Z} g_k \phi(2t - k), \{g_k\}_{k \in Z} \in l^2 \end{cases} \quad (20-4)$$

比较式 (20-3) 和式 (20-4),  $u_0(t)$  和  $u_1(t)$  分别退化为尺度函数  $\phi(t)$  和小波基函数  $\psi(t)$ 。式 (20-3) 是式 (20-1) 的等价表示。把这种等价表示推广到  $n \in Z_+$  (非负整数) 的情况, 即得到式 (20-1) 的等价表示为

$$U_{j+1}^n = U_j^n \oplus U_j^{2n+1}, j \in Z, n \in Z_+ \quad (20-5)$$

定义 (小波包) 由式 (20-2) 构造的序列  $\{u_n(t)\}$  (其中  $n \in Z_+$ ) 称为由基函数  $u_0(t) = \phi(t)$  确定的正交小波包。





由于  $\phi(t)$  由  $h_k$  唯一确定, 所以又称  $\{u_n(t)\}_{n \in \mathbb{Z}}$  为关于序列  $\{h(k)\}$  的正交小波包。

### 20.1.2 小波包的性质

定理 1 设非负整数  $n$  的二进制表示为

$$n = \sum_{i=1}^{\infty} \varepsilon_i 2^{i-1}, \quad \varepsilon_i = 0 \text{ 或 } 1$$

则小波包  $\hat{u}_n(\omega)$  的傅里叶变换由下式给出:

$$\hat{u}_n(\omega) = \prod_{i=1}^{\infty} m_{\varepsilon_i}(\omega/2^i) \quad (20-6)$$

式中,  $m_0(\omega) = H(\omega) = \frac{1}{\sqrt{2}} \sum_{k=-\infty}^{+\infty} h(k) e^{-jk\omega}$ ;  $m_1(\omega) = G(\omega) = \frac{1}{\sqrt{2}} \sum_{k=-\infty}^{+\infty} g(k) e^{-jk\omega}$ 。

定理 2 设  $\{u_n(t)\}_{n \in \mathbb{Z}}$  是正交尺度函数  $\phi(t)$  的正交小波包, 则

$$\langle u_n(t-k), u_n(t-l) \rangle = \delta_{kl}$$

即  $\{u_n(t)\}_{n \in \mathbb{Z}}$  构成  $L^2(R)$  的规范正交基。

### 20.1.3 小波包的空间分解

令  $\{u_n(t)\}_{n \in \mathbb{Z}}$  是关于  $h_k$  的小波包族, 考虑用下列方式生成子空间族。现在令  $n = 0, 1, 2, \dots$ ;  $j = 0, 1, 2, \dots$ , 并对式 (20-1) 进行迭代分解, 则有

$$\begin{aligned} W_j &= U_j^1 = U_{j-1}^2 \oplus U_{j-1}^3 \\ U_{j-1}^2 &= U_{j-2}^4 \oplus U_{j-2}^5, U_{j-1}^3 = U_{j-2}^6 \oplus U_{j-2}^7, \dots \end{aligned}$$

因此, 我们很容易得到小波子空间  $W_j$  的各种分解如下:

$$\begin{aligned} W_j &= U_{j-1}^2 \oplus U_{j-1}^3 \\ W_j &= U_{j-2}^4 \oplus U_{j-2}^5 \oplus U_{j-2}^6 \oplus U_{j-2}^7 \\ &\dots \\ W_j &= U_{j-k}^{2^k} \oplus U_{j-k}^{2^k+1} \oplus \dots \oplus U_{j-k}^{2^{k+1}} \oplus U_{j-k}^{2^{k+1}-1} \\ &\dots \\ W_j &= U_0^{2^j} \oplus U_0^{2^j+1} \oplus \dots \oplus U_0^{2^{j+1}-1} \end{aligned}$$

$W_j$  空间分解的子空间序列可写作  $U_{j-1}^{2^l+m}$ ,  $m = 0, 1, \dots, 2^l-1$ ;  $l = 1, 2, \dots$ 。子空间序列  $U_{j-1}^{2^l+m}$  的标准正交基为  $\{2^{-(j-1)/2} u_{2^l+m}(2^{j-l}t-k) : k \in \mathbb{Z}\}$ 。容易看出, 当  $l = 0$ 、 $m = 0$  时, 子空间序列  $U_{j-1}^{2^l+m}$  简化为  $U_j^1 = W_j$ , 相应的正交基简化为  $2^{-j/2} u_1(2^{-j}t-k) = 2^{-j/2} \psi(2^{-j}t-k)$ , 它恰好是标准正交小波族  $\{\psi_{j,k}(t)\}$ 。



若  $n$  是一个倍频程细划的参数, 即令  $n = 2^l + m$ , 则我们有小波包的简略记号  $\psi_{j,k,n}(t) = 2^{-j/2} \psi_n(2^{-j}t - k)$ , 其中,  $\psi_n(t) = 2^{l/2} u_{2^l+m}(2^l t)$ 。我们把  $\psi_{j,k,n}(t)$  称为既有尺度指标  $j$  位置指标  $k$  和频率指标  $n$  的小波包。将它与前面的小波  $\psi_{j,k}(t)$  做比较可知, 小波只有离散尺度  $j$  和离散平移  $k$  两个参数, 而小波包除了这两个离散参数外, 还增加了一个频率参数  $n = 2^l + m$ 。正是这个频率新参数的作用, 使得小波包克服了小波时间分辨率高时频率分辨率低的缺陷, 于是, 参数  $n$  表示  $\psi_n(t) = 2^{l/2} u_{2^l+m}(2^l t)$  函数的零交叉数目, 也就是其波形的震荡次数。

定义(小波库) 由  $\psi_n(t)$  生成的函数族  $\psi_{j,k,n}(t)$  ( $n \in Z_+$ ;  $j, k \in Z$ ) 称为由尺度函数  $\psi(t)$  构造的小波库。

推论 1 对于每个  $j = 0, 1, 2, \dots$ , 有

$$L^2(R) = \bigoplus_{j \in Z} W_j = \dots \oplus W_{-1} \oplus W_0 \oplus U_0^2 \oplus U_0^3 \oplus \dots \quad (20-7)$$

这时, 族

$$\{u_{j,k}, u_n(t-k) | j = \dots, -1, 0; n = 2, 3, \dots, \text{且 } k \in Z\} \quad (20-8)$$

是  $L^2(R)$  的一个正交基。

随着尺度  $j$  的增大, 相应正交小波基函数的空间分辨率越高, 而其频率分辨率越低, 这正是正交小波基的一大缺陷。而小波包却具有将随  $j$  增大而变宽的频谱窗口进一步分割变细的优良性质, 从而克服了正交小波变换的不足。

小波包可以对  $W_j$  进一步分解, 从而提高频率分辨率, 是一种比多分辨分析更加精细的分解方法, 具有更好的时频特性。

## 20.1.4 小波包算法

下面给出小波包的分解算法和重构算法。

设  $g_j^n(t) \in U_j^n$ , 则  $g_j^n$  可表示为

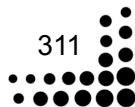
$$g_j^n(t) = \sum_l d_l^{j,n} u_n(2^j t - l) \quad (20-9)$$

小波包分解算法: 由  $\{d_l^{j+1,n}\}$  求  $\{d_l^{j,2n}\}$  与  $\{d_l^{j,2n+1}\}$ , 得

$$\begin{cases} d_l^{j,2n} = \sum_k a_{k-2l} d_k^{j+1,n} \\ d_l^{j,2n+1} = \sum_k b_{k-2l} d_k^{j+1,n} \end{cases} \quad (20-10)$$

小波包重构算法: 由  $\{d_l^{j,2n}\}$  与  $\{d_l^{j,2n+1}\}$  求  $\{d_l^{j+1,n}\}$ , 得

$$d_l^{j+1,n} = \sum_k [h_{l-2k} d_k^{j,2n} + g_{l-2k} d_k^{j,2n+1}]$$







相对于小波变换，小波包变换能够对图像中的高频部分进行分解，具有更强的适应性，因此更加适合图像的各种处理。

小波包分析属于线性时频分析法，它具有良好的时频定位特性以及对信号的自适应能力，因而能够对各种时变信号进行有效的分解。

## 20.2 小波包函数用法

本节将详细讲述关于小波包分析方面的一些函数的用法。它们的主要功能有一维和二维小波包分解、一维和二维小波包重构、最佳（优）小波包基的选择（最优树的选择）等。

### 20.2.1 一维小波包的分解函数

在 MATLAB 中实现一维小波包分解的函数是 `wpdec`，其调用格式有以下两种：

```
T=wpdec(X,N,'wname',E,P)
T=wpdec(X,N,'wname')
```

说明：`wpdec` 是一个一维的小波包分解函数。

对于格式 `T=wpdec(X,N,'wname',E,P)`，它根据小波包函数、熵标准 `E` 和参数 `P` 对信号 `X` 进行小 `N` 层小波包分解，并返回小波包分解结构 `[T,D]`（`T` 为树结构，`D` 为数据结构）。其中，`E` 用来指定熵标准，`E` 的类型可以是 `shannon`、`threshold`、`norm`、`log energy`、`sure` 或 `user`；`P` 是一个可选的参数，它的选择根据参数 `E` 的值来决定。

对于格式 `T=wpdec(X,N,'wname','shannon')`，默认为 `shannon` 熵标准。

小波包分析是多分辨分析的推广，它提供了更为丰富和精确的信号分析方法。小波包元素是由 3 个参数来确定的一个波形，这 3 个最基本参数为位置、尺度和频率。

在小波包分解中，每个高频系数向量也像低频部分的分解一样，被分解成两部分，因而，它提供了更丰富的分析方法。在一维情况下，它产生了一个完整的二叉树；在二维情况下，它产生了一个完整的四叉树。

【例 20-1】`wpdec` 函数应用实例。

```
% 装入信号
load noisdopp;
x=noisdopp;
% 用 db1 小波包分解信号 x 到第 3 层
% 采用 shannon 熵的标准
t=wpdec(x,3,'db1','shannon');
plot(t);% 画树结构的图形
```

程序运行结果如图 20-2 所示。

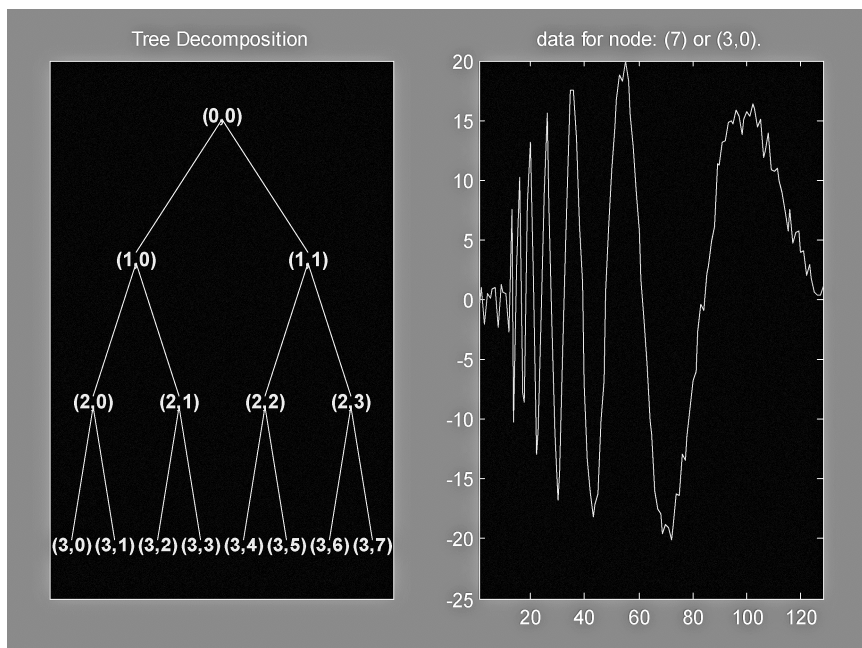


图 20-2 小波包分解树结构

### 20.2.2 一维小波包的重构函数

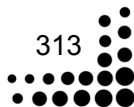
在 MATLAB 中实现一维小波包重构的函数是 `wprec`，其调用格式如下：

```
X=wprec(T)
```

`wprec` 是一个一维的小波包分解函数。它对小波包的分解结构 `T` 进行重构，并返回重构后的向量 `X`。其中，`T` 是树结构。

【例 20-2】`wprec` 函数应用实例。

```
% 装入信号
load noisdopp;
x=noisdopp(1:1000);
figure(1);
subplot(211);
plot(x);
title('原始信号');
% 用 db1 小波包分解信号 x 到第 3 层
% 采用 shannon 熵的标准
t=wpdec(x,3,'db1','shannon');
plot(t);% 画小波包树结构的图形
recx=wprec(t);% 重构小波包分解结构 T
figure(1);
subplot(212);plot(recx);
title('重构后的信号');
```





程序运行结果如图 20-3 和图 20-4 所示。

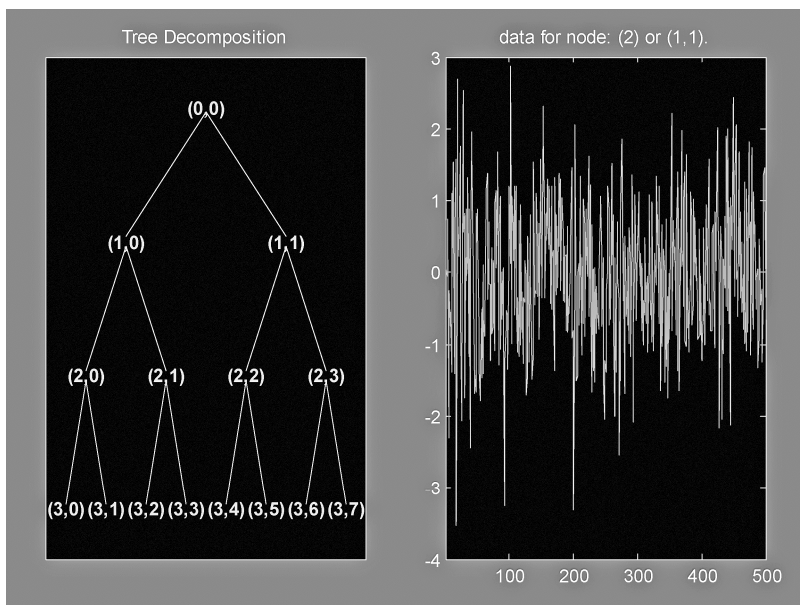


图 20-3 小波包分解树结构

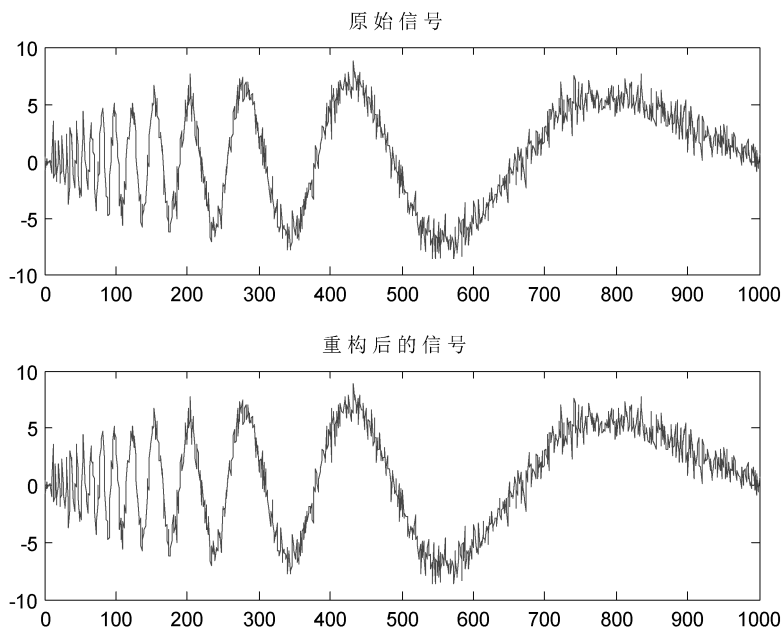


图 20-4 noisdopp 原始信号与重构后的信号

### 20.2.3 二维小波包的分解函数

在 MATLAB 中实现二维小波包分解的函数是 `wpdec2`，其调用格式有以下两种：



```
T=wpdec2(X,N,'wname',E,P)
```

```
T=wpdec2(X,N,'wname')
```

wpdec2 是一个二维小波包分解函数。

**格式** 根据相应的小波包分解向量  $X$  和指定的小波对  $X$  进行  $N$  层小波包分解,并返回树结构  $T$ 。其中, $E$  是一个字符串,用来指定熵类型, $E$  的类型可以是 shannon、threshold、norm、log energy、sure 或 user; $P$  是一个可选的参数,它的选择根据参数  $E$  的值来决定。

$T=wpdec2(X,N,'wname')$ 与  $T=wpdec2(X,N,'wname','shannon')$ 是等价的。

**【例 20-3】**wpdec2 函数应用实例。

```
% 现在的扩展模式为零扩展
% 装入图像,X 包含装入的图像
load tire;
% 用默认的 shannon 熵分解图像
t=wpdec2(X,2,'db1');
plot(t);% 画四叉树结构图
```

程序运行结果如图 20-5 所示。

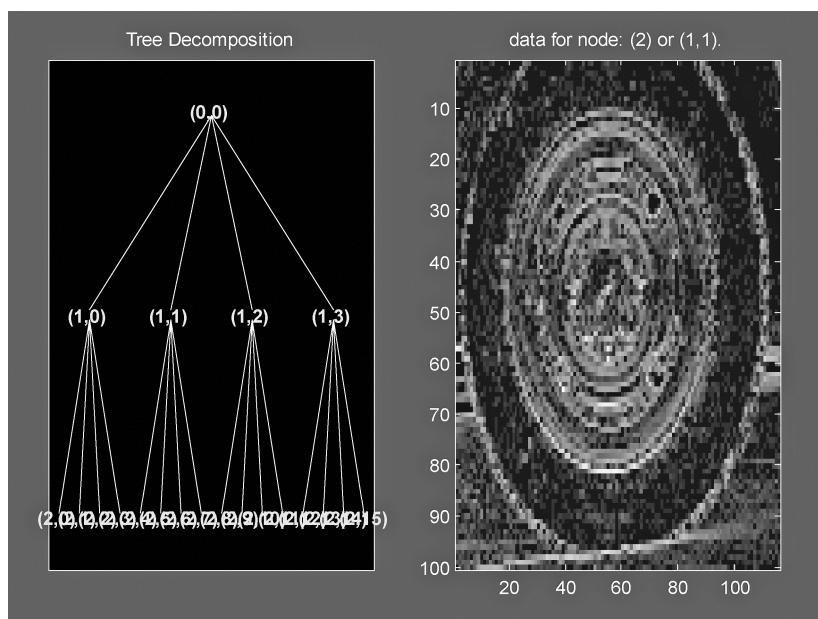


图 20-5 小波包分解树结构

## 20.2.4 二维小波包的重构函数

在 MATLAB 中实现二维小波包重构的函数是 wprec2,其调用格式如下:

```
X=wprec2(T)
```

wprec2 是一个二维的小波包分析函数。它对小波包的分解结构  $T$  进行重构,并返回重构后的向量  $X$ 。其中, $T$  是树结构。



## 【例 20-4】wprec2 函数应用实例。

```
% 装入图像，X 包含装入的图像
load tire;
figure(1);
subplot(221);
image(X);
colormap(map);
title('原始图像');
axis square;
% 用默认的 shannon 熵分解图像
t=wpdec2(X,2,'db1');
plot(t);% 画四叉树结构图
% 对分解结构[t,d]进行重构
rectire=wprec2(t);
% 画出重构后的图像
figure(1);
subplot(222);
image(rectire);
colormap(map);
title('重构后的图像');
axis square;
```

程序运行结果如图 20-6 和图 20-7 所示。

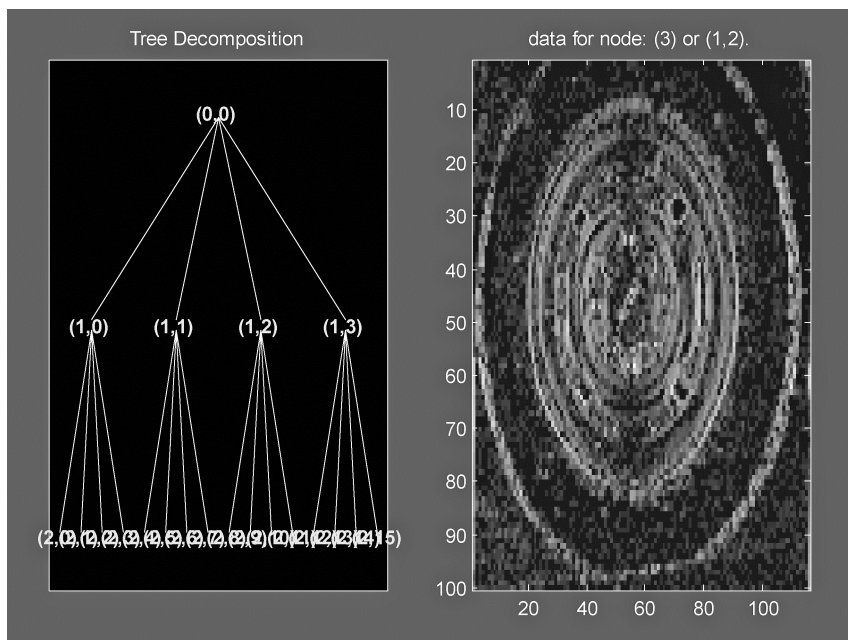


图 20-6 小波包分解树结构

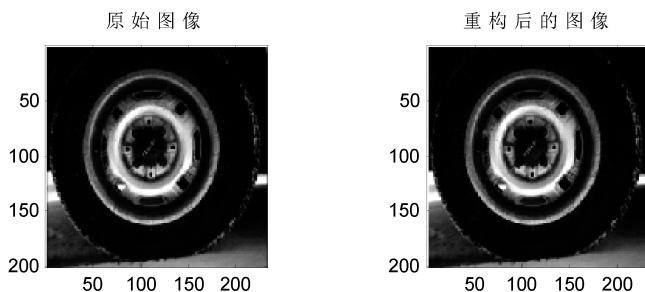


图 20-7 tire 原始信号与重构后的信号

### 20.2.5 重新组合小波包函数

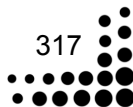
在 MATLAB 中实现重新组合小波包的函数是 `wpjoin`，其调用格式有以下 4 种：

```
T=wpjoin(T,N)
[T,X]=wpjoin(T,N)
[T]=wpjoin(T)
[T,X]=wpjoin(T)
```

`wpjoin` 是一个一维或二维的小波包分析函数，它用来重新组合小波包，即把节点  $N$  以下的二叉树去掉后，返回一个更新后的小波包分解结构。节点从左到右、从上到下进行计算，根节点的索引为零。

**【例 20-5】** `wpjoin` 函数应用实例。

```
% 装入信号
load noisdopp;
x=noisdopp
figure(1);
subplot(121);
plot(x);
title('原始信号');
% 用 db1 小波包对信号 x 分解到第 3 层
wpt=wpdec(x,3,'db1');
plot(wpt);% 画小波包树结构的图形
% 重组小波包 (1,1) 或第 2 个节点，并返回更新后的小波包分解结构
% 参数 c 是节点 2 的小波包分解系数
[wpt,c]=wpjoin(wpt,2); % 2 是指节点 2，也可以用[1,1]表示
plot(wpt);% 画小波包树结构的图形
% 以图形的方式显示分解系数的大小
figure(1);
subplot(122);
plot(c);
title('节点 2 的小波包分解系数');
```





程序运行结果如图 20-8 ~ 图 20-10 所示。

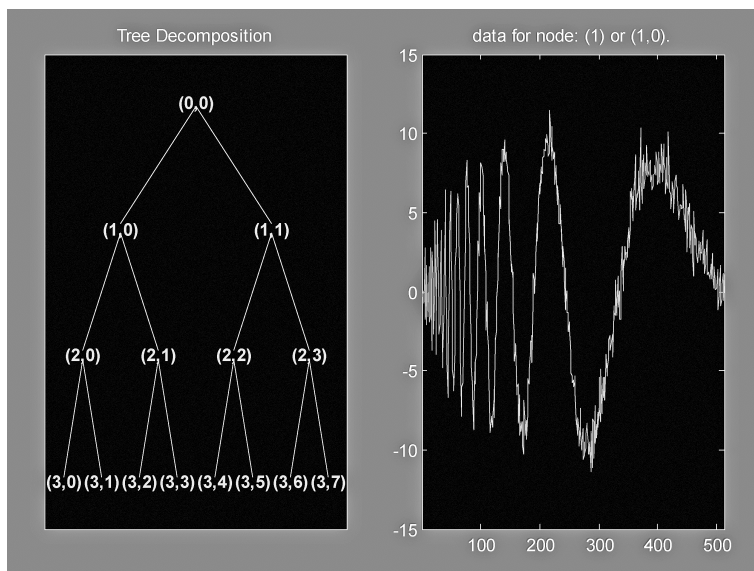


图 20-8 小波包分解树结构

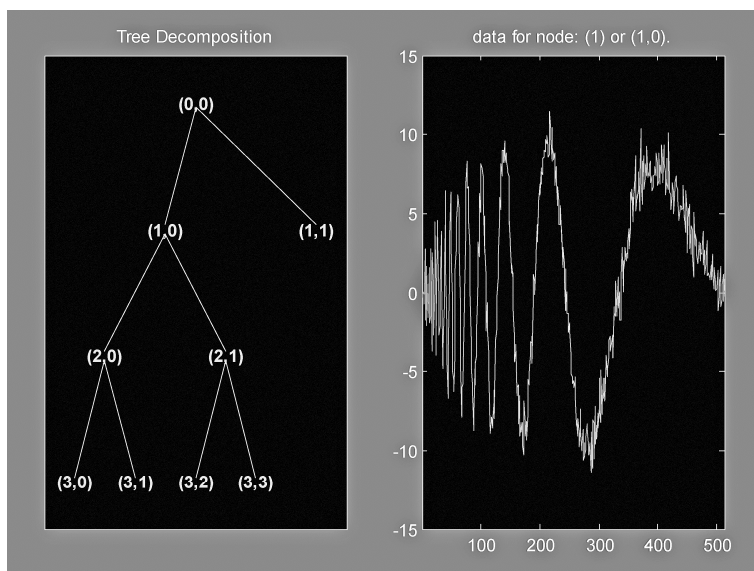


图 20-9 `wpjoin` 函数应用实例

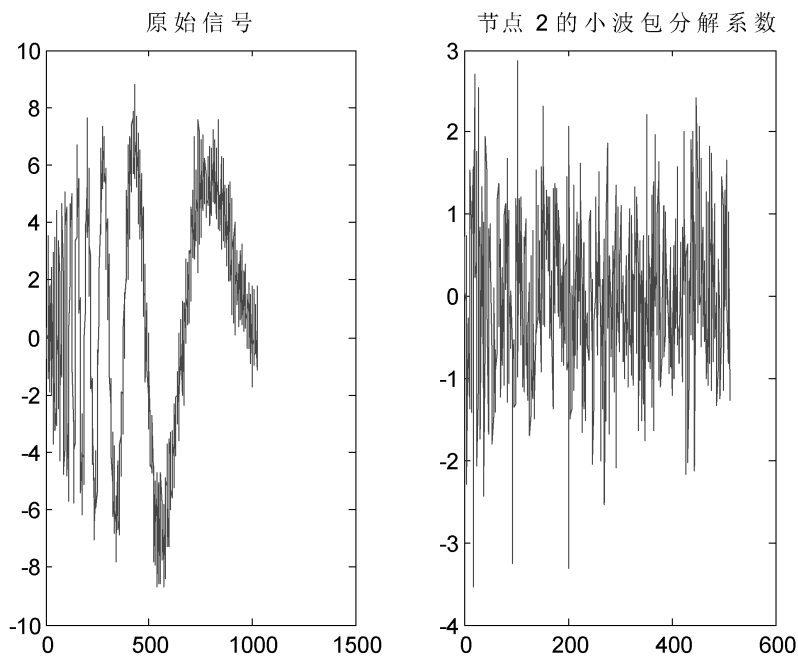


图 20-10 节点 2 的小波包分解系数

### 20.2.6 计算最佳树函数

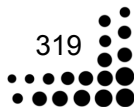
在 MATLAB 中实现计算最佳树的函数是 `besttree`，其调用格式有以下 3 种：

```
TT=besttree(T)
[TT,E]=besttree(T)
[TT,E,N]=besttree(T)
```

`besttree` 是一个一维或二维的小波包分析函数，它能根据一个熵标准来计算初始树的最佳子树，算出的子树比初始树小得多。

**【例 20-6】**`besttree` 函数应用实例。

```
% 装入信号
load noisdopp;
x=noisdopp
% 用 db1 小波包对信号 x 分解到第 3 层
wpt=wpdec(x,3,'db1');
% 用默认的 (shannon) 熵分解小波包[3, 0]
wpt=wpsplt(wpt,[3,0]);
% 画小波包树的结构 wpt
plot(wpt);
% 计算最佳树
[bt,e,n]=besttree(wpt);
% 画出最佳树的结构 bt
```







```
plot(bt);  
disp('表示初始树的各个节点的熵值向量 e 为 :')  
e  
disp('表示初始树的各个节点索引序号向量 n 为 :')  
n
```

程序运行结果如图 20-11 和图 20-12 所示。

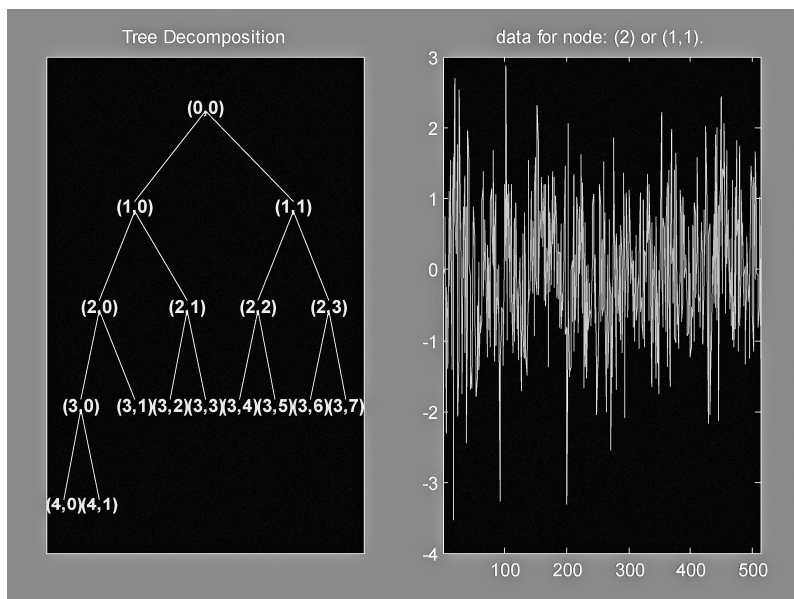


图 20-11 初始小波包分解树结构

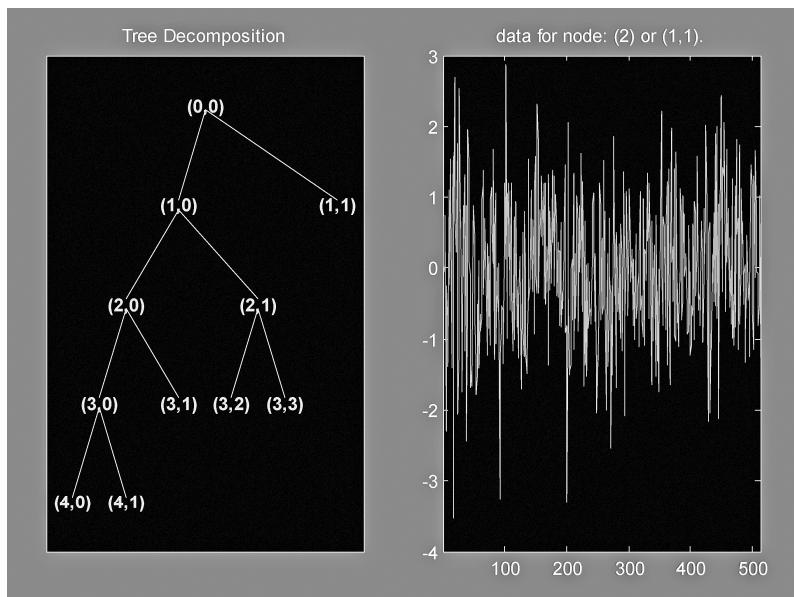


图 20-12 最佳树结构



表示初始树的各个节点的熵值向量  $e$  为

```
e =
    1.0e+004 *
    -9.8173
    -9.7822
    -0.0350
    -9.7303
    -0.0519
    -9.5880
    -0.1423
    -0.0318
    -0.0200
    -9.3112
    -0.2768
```

表示初始树的各个节点索引序号向量  $n$  为：

```
n =
     2
```

从图 20-11 和图 20-12 可以看出，计算出的最佳子树比初始树要小得多。由于初始树中，只有索引为 2 的节点以下的二叉树被合并，所以  $n$  为一个单元素向量。

### 20.2.7 小波包分析函数

在 MATLAB 中实现小波包分析的函数是 `wpfun`，其调用格式有以下两种：

```
[WPWS,X]=wpfun('wname',NUM,PREC)
[WPWS,X]=wpfun('wname',NUM)
```

`wpfun` 是一个小波包分析函数。格式 计算指定的小波 `wname` 的小波包，且时间长度为二进制时间间隔。输出矩阵 `WPWS` 包含  $0 \sim \text{NUM}$  的小波包函数，且按  $[W_0; W_1; \dots; W_{\text{NUM}}]$  的顺序排列，输出向量  $X$  是普通的网格矢量。

**【例 20-7】** `wpfun` 函数应用实例。

```
% 计算 db2 的小波包函数 W(n), n=0,1,...,7
[wp,x]=wpfun('db2',7);
for i=1:8
    w=wp(i,:);
    subplot(4,2,i);plot(w);
    ylabel(['W',num2str(i)]);
end
```

程序运行结果如图 20-13 所示。

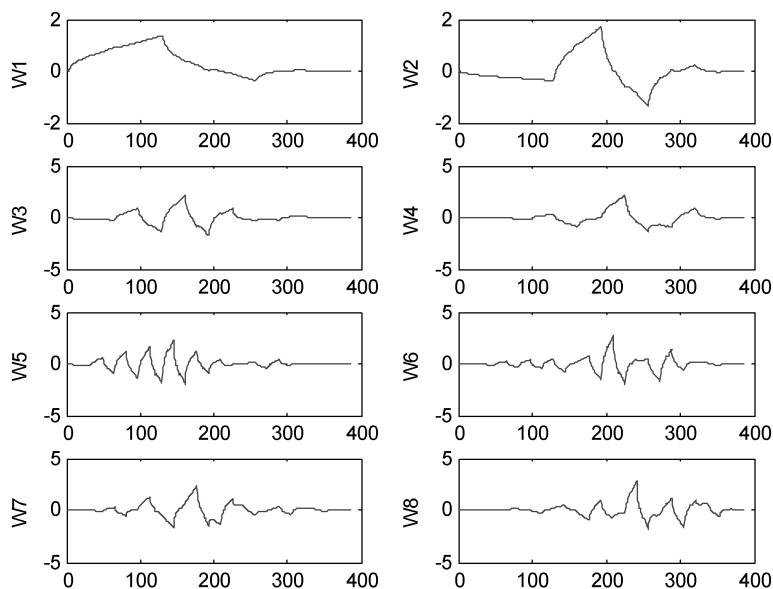


图 20-13 wfun 函数应用实例

## 20.2.8 更新小波包熵值函数

在 MATLAB 中实现更新小波包熵值的函数是 `entrupd`，其调用格式有以下两种：

```
E=entrupd(T,ENT)
E=entrupd(T,ENT,PAR)
```

`entrupd` 函数是一个一维或二维的小波包应用函数。

【例 20-8】`entrupd` 函数应用实例。

```
% 装入信号
load noisdopp;
x=noisdopp
% 用 db2 小波包对信号 x 分解到第 2 层，用 shannon 熵作为熵标准
t=wpdec(x,2,'db1','shannon');
% 计算小波包分解树的节点
nodes=allnodes(t);
% 读取初始小波包分解数据结构中各节点的熵值
disp('初始各节点的熵值：');
ent=read(t,'ent',nodes)
% 不改变树结构和数据结构，选用'threshold'熵标准（阈值为 0.5），更新节点的熵值
d=entrupd(t,'threshold',0.5);
% 读取更新后小波包分解数据结构中各节点的熵值
disp('更新后各节点的熵值：');
nent=read(t,'ent')
```

程序运行结果如图 20-14 所示。

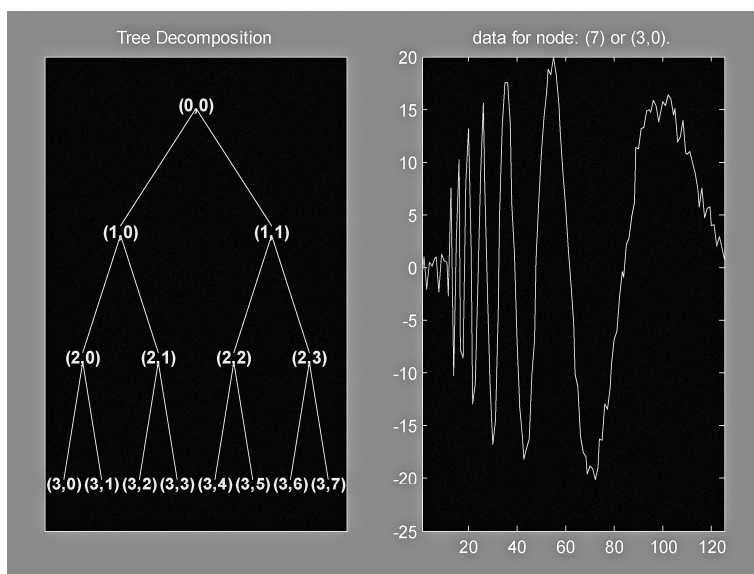


图 20-14 小波包分解树结构

初始各节点的熵值为

```
ent =
1.0e+004 *
-5.8615
-6.8204
-0.0350
-7.7901
-0.0497
-0.0205
-0.0138
```

更新后各节点的熵值为

```
nent =
1.0e+004 *
-5.8615
-6.8204
-0.0350
-7.7901
-0.0497
-0.0205
-0.0138
```

### 20.2.9 计算小波包熵函数

在 MATLAB 中实现计算小波包熵的函数是 `wentropy`，其调用格式有以下两种：



```
E=wentropy(X,T,P)
```

```
E=wentropy(X,T)
```

说明：wentropy 是一个一维或二维的小波包分析函数。

【例 20-9】wentropy 函数应用实例。

```
x=rand(1,200);          % 产生一个初始的随机信号
% 计算信号 x 的 shannon 熵
e1=wentropy(x,'shannon')
% 计算信号 x 的对数能量熵
e2=wentropy(x,'log energy')
% 计算信号 x 的阈值熵，阈值等于 0.2
e3=wentropy(x,'threshold',0.2)
% 计算信号 x 的 SURE 熵，阈值等于 3
e4=wentropy(x,'sure',3)
% 计算信号 x 的范数熵，范数指数等于 1.1
e5=wentropy(x,'norm',1.1)
```

程序运行结果如下：

```
e1 =
    48.8337
e2 =
   -353.2800
e3 =
    168
e4 =
   -131.0999
e5 =
    99.6449
```

## 20.2.10 分割小波包函数

在 MATLAB 中实现分割小波包的函数是 wpsplt，其调用格式有以下 3 种：

```
T=wpsplt(T,N)
```

```
[T,CA,CD]=wpsplt(T,N)
```

```
[T,CA,CH,CV,CD]=wpsplt(T,N)
```

说明：wpsplt 是一个一维或二维小波包分析函数。它在重组一个节点后，更新树结构和数据结构。

【例 20-10】wpsplt 函数应用实例。

```
% 装入信号
load noisdopp;
x=noisdopp(1:1000);
% 用 db1 小波包分解信号 x 到第 3 层
```



```
% 采用 shannon 熵的标准
t=wpdec(x,3,'db1','shannon');
plot(t);          % 画出小波包树结构的图形
% 重新分解小波包节点 (3,0) 或第 7 个节点
wpt=wpsplt(t,[3,0]);
% 或等价于 wpt=wpsplt(t,7);
plot(wpt);        % 画出小波包树结构的图形
```

程序运行结果如图 20-15 所示。

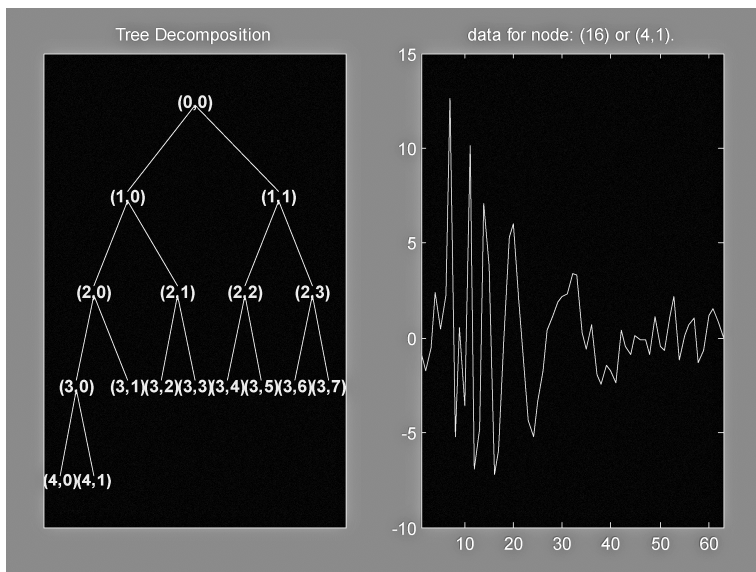


图 20-15 重新分解小波包节点 (3,0)

### 20.2.11 计算完整最佳小波包树函数

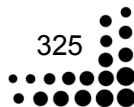
在 MATLAB 中实现计算完整最佳小波包树的函数是 `bestlevt`，其调用格式有以下两种：

```
TT= bestlevt(T)
[TT,E]= bestlevt(T)
```

`bestlevt` 是一个一维或二维的小波包分析函数，它可以根据一种熵标准计算出初始树的最佳完整子树，这个完整的子树比初始树的深度小一些。

【例 20-11】`bestlevt` 函数应用实例。

```
% 装入信号
load noisdopp;
x=noisdopp
% 用 db1 小波包对信号 x 分解到第 3 层
wpt=wpdec(x,3,'db1');
% 用默认的 (shannon)熵分解小波包[3,0]
wpt=wpsplt(wpt,[3,0]);
```





```
% 画小波包树的结构 wpt 的图形
plot(wpt);
% 计算最佳层次的树
blt=bestlevt(wpt);
% 画出最佳层次的树结构 blt 的图形
plot(blt);
```

程序运行结果如图 20-16 和图 20-17 所示。

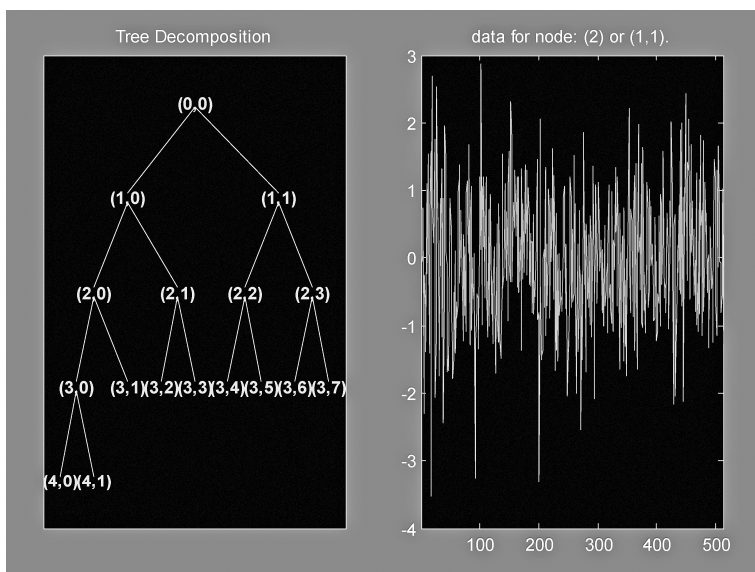


图 20-16 初始小波包分解树结构

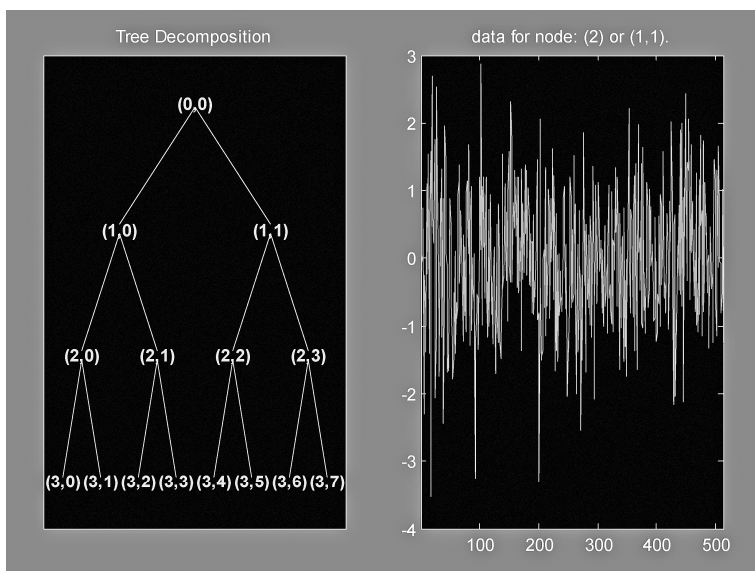


图 20-17 最佳层次的树结构



### 20.2.12 从小波包树中提取小波树函数

在 MATLAB 中实现从小波包树中提取小波树的函数是 `wp2wtree`，其调用格式如下：

```
TT=wp2wtree(T)
```

`wp2wtree` 是一个一维或二维小波包分析函数，它根据小波包分解树修改计算树结构 `T`，并返回小波树 `T`。

【例 20-12】`wp2wtree` 函数应用实例。

```
% 装入信号
load noisdopp;
x=noisdopp
% 用 db1 小波包对信号 x 分解到第 3 层
wpt=wpdec(x,3,'db1');
% 画小波包树的结构 wpt 的图形
plot(wpt);
% 计算小波包树
wt=wp2wtree(wpt);
% 画小波包树的结构 wt 的图形
plot(wt);
```

程序运行结果如图 20-18 和图 20-19 所示。

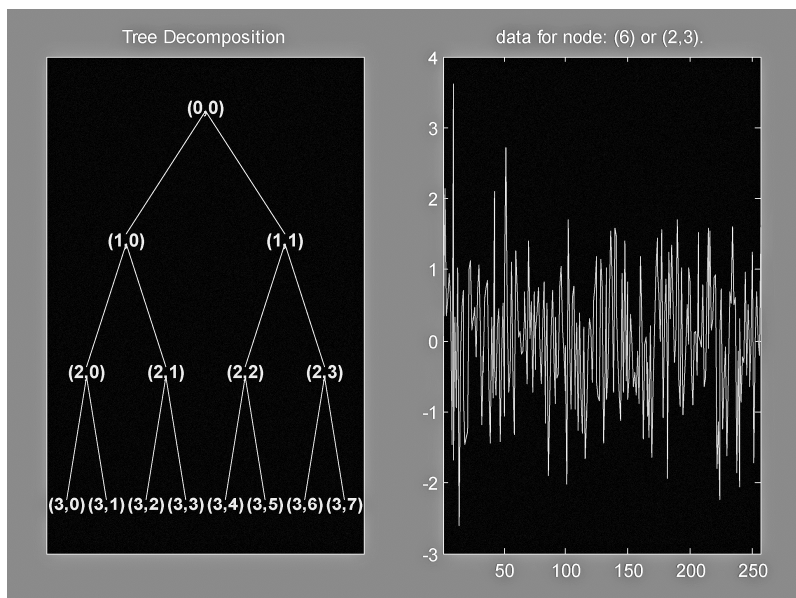
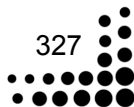


图 20-18 初始小波包分解树结构





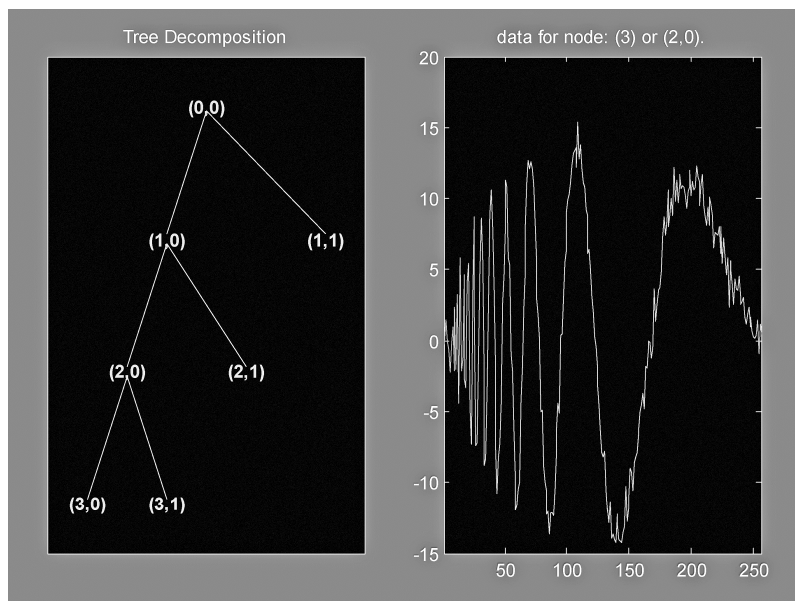


图 20-19 提取小波树

### 20.2.13 剪切小波包分解树函数

在 MATLAB 中实现剪切小波包分解树的函数是 `wpcutree`，其调用格式有以下两种：

```
T=wpcutree(T,L)
[T,RN]=wpcutree(T,L)
```

`wpcutree` 是一个一维或二维的小波包分析函数，它将小波包分解树第  $L$  层以下的所有二叉子树全部剪掉，并返回经过修改计算后的小波包分解结构  $T$ 。其中， $L$  是小波包分解树的层数。

【例 20-13】`wpcutree` 函数应用实例。

```
% 装入信号
load noisdopp;
x=noisdopp
% 用 db1 小波包对信号 x 分解到第 3 层
wpt=wpdec(x,3,'db1');
plot(wpt); % 画小波包树结构的图形
% 在第 2 层剪切小波包树
[nwpt,m]=wpcutree(wpt,2); % 2 是指第 2 层
plot(nwpt); % 画小波包树结构 (nwpt) 的图形
disp('初始树中经过重组节点的索引序号：')
m
```

程序运行结果如图 20-20 和图 20-21 所示。

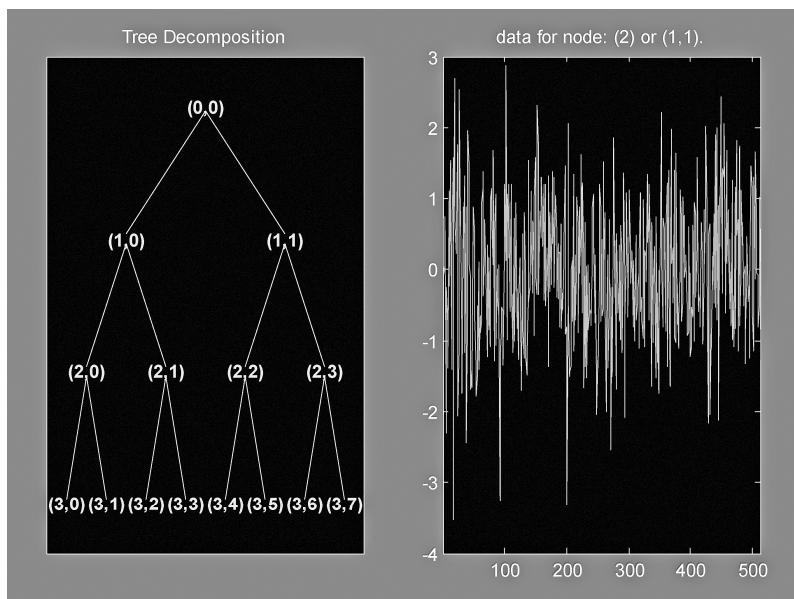


图 20-20 初始小波包分解树结构

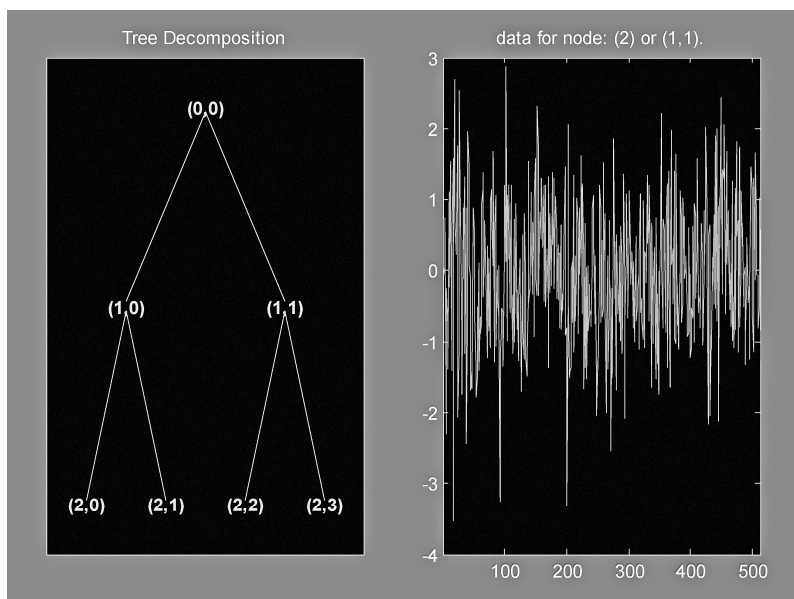


图 20-21 剪切后的小波包分解树

初始树中经过重组节点的索引序号为：

m =

- 3
- 4
- 5
- 6



## 20.2.14 计算小波包系数函数

在 MATLAB 中实现计算小波包系数的函数是 `wpcoef`，其调用格式如下：

```
X=wpcoef(T,N)
```

`wpcoef` 是一个一维或二维的小波包分析函数。它返回与树 `T` 节点 `N` 对应的重构系数。其中，`T` 是树结构。

【例 20-14】`wpcoef` 函数应用实例。

```
% 装入信号
load noisdopp;
x=noisdopp;
figure(1);
subplot(221);
plot(x);
title('原始信号');
% 用 db1 小波包分解信号 x 到第 3 层
t=wpdec(x,3,'db1','shannon');
plot(t);% 画树结构图形
% 读取小波包 (2,1) 的系数
cfs=wpcoef(t,[2,1]);
figure(1);
subplot(222);
plot(cfs);
title('小波包 (2,1) 的系数');
```

程序运行结果如图 20-22 和图 20-23 所示。

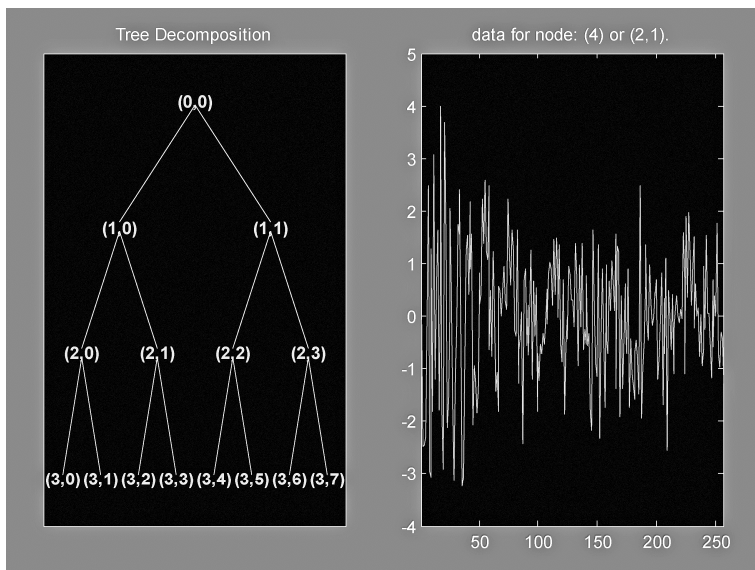


图 20-22 小波包分解树结构

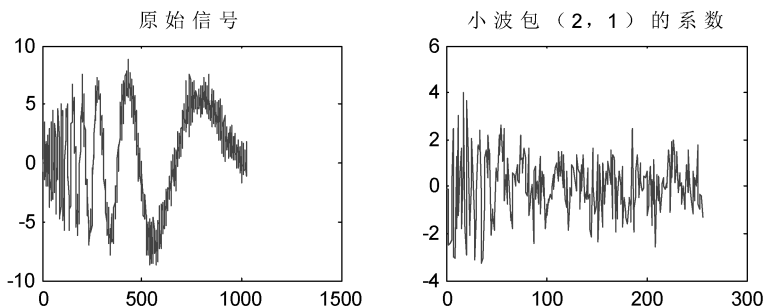


图 20-23 Noisdopp 原信号与分解图

### 20.2.15 小波包分解系数的重构函数

在 MATLAB 中实现小波包分解系数重构的函数是 `wprcoef`，其调用格式如下：

```
X=wprcoef(T,N)
```

`wprcoef` 是一个一维或二维的小波包分析函数。它计算节点  $N$  的小波包分解系数的重构信号。其中， $T$  是树结构。

**【例 20-15】** `wprcoef` 函数应用实例。

```
% 装入信号
load noisdopp;
x=noisdopp(1:1000);
figure(1);
subplot(121);
plot(x);
title('原始信号');
% 用 db1 小波包分解信号 x 到第 3 层
t=wpdec(x,3,'db1','shannon');
plot(t);% 画树结构的图形
% 重构小波包的节点 (2, 1)
rcfs=wprcoef(t,[2,1]);
figure(1);
subplot(122);
plot(rcfs);
title('重构的小波包节点 (2, 1)');
```

程序运行结果如图 20-24 和图 20-25 所示。

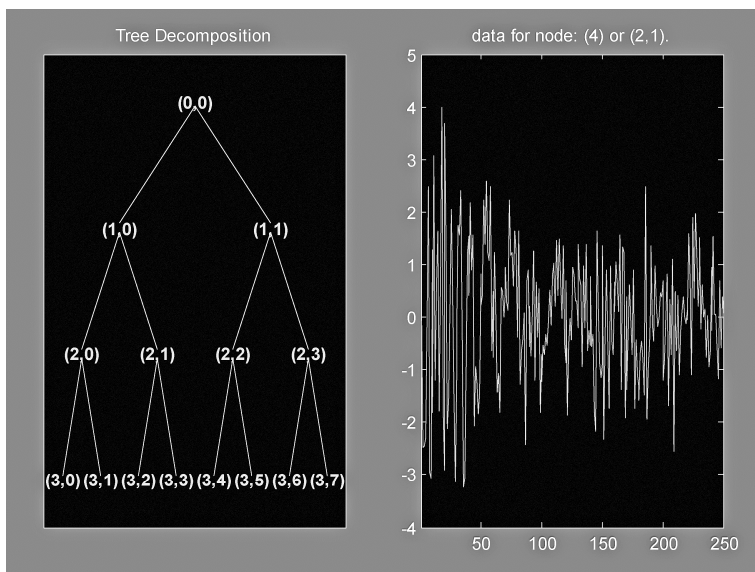


图 20-24 小波包分解树结构

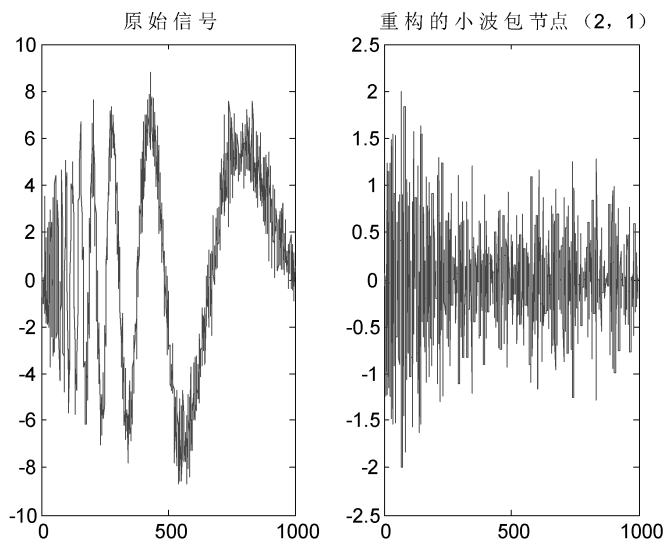


图 20-25 Noisdopp 原信号与分解图

【例 20-16】对信号进行小波包分解后，观察其树结构。

```
%读信号
load noisbump
x = noisbump;
%3 层小波包分解
t = wpdec(x,3,'db2');
%显示小波包树结构
fig = plot(t);
```

计算得到的小波包树结构如图 20-26 所示。

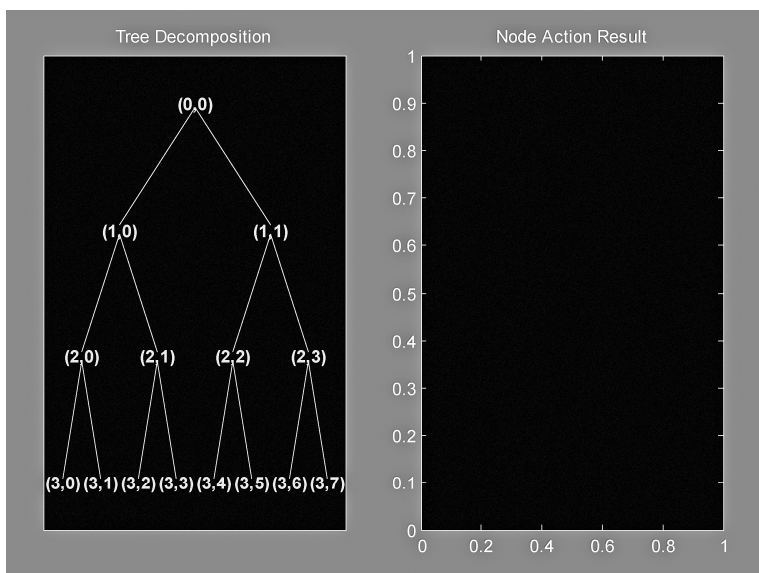


图 20-26 3 层小波包树结构

在菜单“Node Label”下选择“Index”，然后单击左边的节点“(9)”，显示其信号波形，得到的结果如图 20-27 所示。

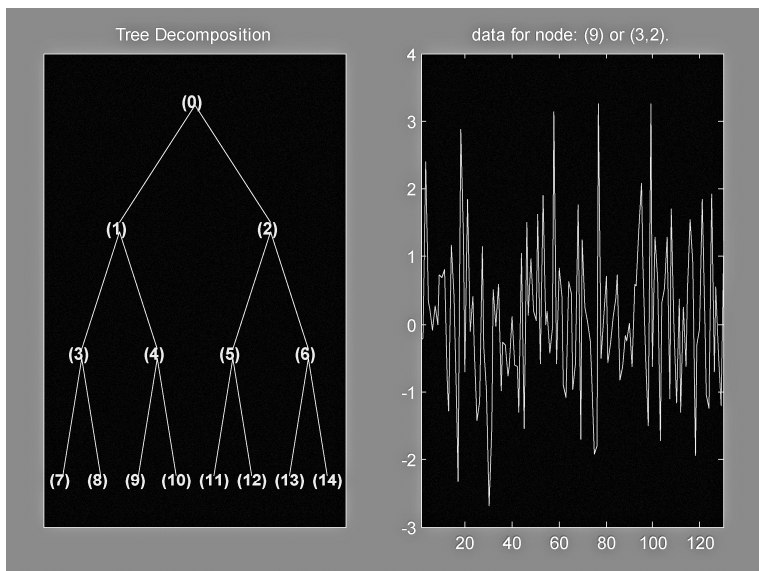


图 20-27 显示节点的信号波形

在菜单“Node Action”下选择“Split/Merge”，然后单击节点“(6)”，合并节点 6；接着在菜单“Node Action”下选择“Visualize”，再单击节点“(6)”，以显示其信号波形，如图 20-28 所示。

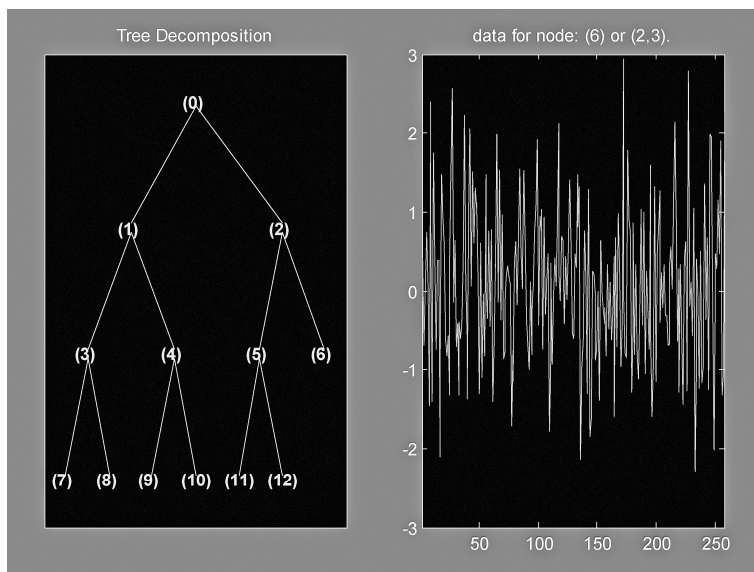


图 20-28 合并节点的信号波形

得到新的树结构后，可以对其进行各种操作。

```
%获取新的树  
newt=plot(t,'read',fig);  
%从命令行修改新树  
newt=wpjoin(newt,4);  
%并显示  
fig2=plot(newt);
```

运行结果如图 20-29 所示，右边显示的是节点 6 处的信号波形。

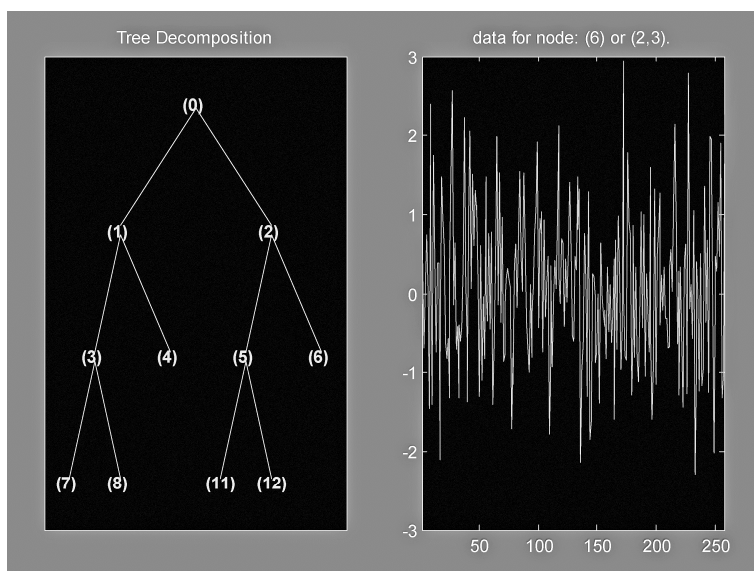


图 20-29 新小波包树的结构

%查看颜色表示的小波包系数

wpviewcf(t,4);

运行结果如图 20-30 所示。

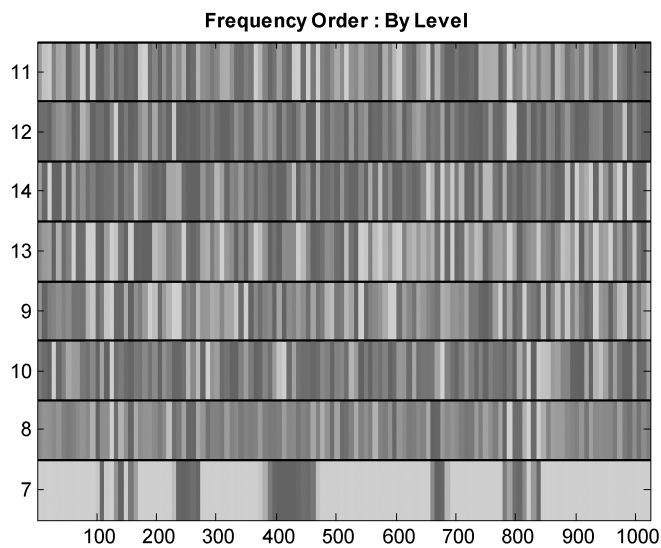


图 20-30 颜色表示的小波包系数



## 第 21 章 多分辨分析及 Mallat 算法分析

信号处理的任务之一是认识客观世界中存在的信号的本质特征,并找出规律。“横看成岭侧成峰,远近高低各不同”,从不同的角度去认识、分析信号有助于了解信号的本质特征。信号最初是以时间(空间)的形式来表达的。除了时间以外,频率是一种表示信号特征最重要的方式。频率的表示方法是建立在傅里叶分析(Fourier Analysis)基础之上的,由于傅里叶分析是一种全局的变换,要么完全在时间域,要么完全在频率域,因此无法表述信号的时频局部性质,而时频局部性质恰好是非平稳信号最基本和最关键的性质。小波变换是一种时间-尺度分析方法,而且在时间、尺度(频率)两域都具有表征信号局部特征的能力,在低频部分具有较高的频率分辨率和较低的时间分辨率,在高频部分具有较高的时间分辨率和较低的频率分辨率,很适合于探测正常信号中夹带的瞬间反常现象并展示其成分。所以,小波变换被称为分析信号的显微镜。小波变换不会“一叶障目,不见泰山”,而是可以做到“管中窥豹,略见一斑”。但是,小波分析不能完全取代傅里叶分析,小波分析是傅里叶分析的发展,而时频分析是一种非线性二次变换,与线性的小波变换相去甚远。下面将研究小波变换的基本理论,为后续的分析研究打下基本理论基础。

### 21.1 小波分析的基本理论

为了分析和处理非平稳信号,在傅里叶分析理论基础上,提出并发展了一系列新的信号分析理论:短时傅里叶变换(Short Time Fourier Transform)或加窗傅里叶变换(Windowed Fourier Transform)、Gabor 变换、时频分析、小波变换、分数阶傅里叶变换(Fractional Fourier Transform)、线调频小波变换等。

短时傅里叶变换是一种单一分辨率的信号分析方法,它的思想是:选择一个时频局部化的窗函数,假定分析窗函数  $g(t)$  在一个短时间间隔内是平稳(伪平稳)的,移动窗函数,使  $f(t)g(t)$  在不同的有限时间宽度内是平稳信号,从而计算出各个不同时刻的功率谱。短时傅里叶变换使用一个固定的窗函数,窗函数一旦确定,其形状就不再发生改变,短时傅里叶变换的分辨率也就确定了。如果要改变分辨率,则需要重新选择窗函数。短时傅里叶变换用来分析分段平稳信号或者近似平稳信号尚可,但是对于非平稳信号,当信号变化剧烈时,就要求窗函数有较高的时间分辨率;而波形变化比较平缓的时刻,主要是低频信号,则要求窗函数有较高的频率分辨率。短时傅里叶变换不能兼顾频率与时间分辨率的需求。短时傅里叶变换窗函数受到 W.Heisenberg 不确定准则的限制,时频窗的面积不小于 2。这也从侧面说明了短时傅里叶变换窗函数的时间与频率分辨率不能同时达到最优。



Gabor 变换是海森伯不确定准则下的最优短时傅里叶变换。高斯窗函数是短时傅里叶变换同时追求时间分辨率与频率分辨率时的最优窗函数。具有高斯窗函数的短时傅里叶变换就是 Gabor 变换。与短时傅里叶变换一样, Gabor 变换也是单一分辨率的。小波变换使用一个窗函数(小波函数), 时频窗面积不变, 但形状可改变。小波函数根据需要调整时间与频率分辨率, 具有多分辨率分析(multiresolution analysis)的特点, 克服了短时傅里叶变换分析非平稳信号单一分辨率的困难。

## 21.2 连续小波变换

在小波分析中, 主要讨论的函数空间为  $L^2(R)$ 。 $L^2(R)$  指  $R$  上平方可积函数构成的函数空间, 即

$$f(t) \in L^2(R) \Leftrightarrow \int_R |f(t)|^2 dt < +\infty$$

若  $f(t) \in L^2(R)$ , 则称  $f(t)$  为能量有限的信号。 $L^2(R)$  也常称能量有限的信号空间。

如果  $\psi(t) \in L^2(R)$ , 其傅里叶变换为  $\hat{\psi}(\omega)$  满足容许性条件(admissible condition)

$$C_\psi = \int_{-\infty}^{+\infty} |\omega|^{-1} |\hat{\psi}(\omega)|^2 d\omega < +\infty \quad (21-1)$$

即  $C_\psi$  有界, 则称  $\psi$  为一个基小波或母小波(mother wavelet)。将母小波经过伸缩和平移后, 就可以得到一个小波序列

$$\psi_{a,b}(t) = |a|^{-1/2} \psi\left(\frac{t-b}{a}\right) \quad (21-2)$$

其中,  $a, b \in R$ , 且  $a \neq 0$ , 称  $a$  为伸缩因子,  $b$  为平移因子。定义

$$(W_\psi f)(a, b) = \langle f, \psi_{a,b} \rangle = |a|^{-1/2} \int_{-\infty}^{+\infty} f(t) \overline{\psi\left(\frac{t-b}{a}\right)} dt$$

为关于基小波  $\psi$  的连续小波变换(或积分小波变换)。其中,  $\bar{\cdot}$  表示对  $X$  的共扼运算。显然, 变换后的函数是二维的, 即小波变换把原来的一维信号变换为二维信号, 以便分析信号(或函数)的时-频特性。而下面的变换

$$f(t) = \frac{1}{C_\psi} \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} [(W_\psi f)(a, b)] \psi_{a,b}(x) \frac{da}{a^2} db$$

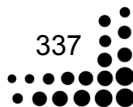
为关于基小波  $\psi$  的逆小波变换。小波逆变换把二维信号重构回原来的一维信号。

从小波的容许性条件式(21-1)可以知道, 母小波  $\psi(t)$  是一个振荡且能量有限的函数, 并且在时域上是快速衰减的, 容易推出

$$\int_{-\infty}^{+\infty} \psi(t) dt = 0 \quad (21-3)$$

即  $\hat{\psi}(0) = 0$ 。这是因为, 若  $\hat{\psi}(0) \neq 0$ , 则当  $\omega = 0$  时,  $|\omega|^{-1} |\hat{\psi}(\omega)|^2 = \infty$ , 即  $C_\psi$  无界。

小波变换的实质在于将  $L^2(R)$  空间中的任意函数  $f(t)$  表示成为其在具有不同伸缩因子  $a$  和平移因子  $b$  的  $\psi_{a,b}(t)$  之上的投影的叠加。与傅里叶变换(仅将  $f(t)$  投影到频率域)不同的是, 小波变换将一维时域函数映射到二维“时间-尺度”域上, 因此  $f(t)$  在小波基上的展开具有多分辨率的特性。通过调整伸缩因子  $a$  和平移因子  $b$ , 可以得到具有不同时-频宽度的小波,





以匹配原始信号的任意位置，达到对信号的时-频局部化分析的目的。

## 21.3 离散小波变换

连续小波变换中的尺度因子和平移因子都是连续变化的实数，在应用中需要计算连续积分，在处理数字信号时很不方便，主要用于理论分析与论证。在实际问题的数值计算中常采用离散形式，即离散小波变换（DWT）。DWT 可以通过离散化 CWT 中的尺度因子  $a$  和平移因子  $b$  得到。通常取

$$a = a_0^m, b = nb_0 a_0^m, m, n \in Z$$

把其带入式 (21-2)，可以得到

$$\psi_{m,n}(t) = |a_0|^{-m/2} \psi(a_0^{-m}t - nb_0), m, n \in Z$$

这时的小波函数就是离散小波。相应的离散小波变换为

$$(W_\psi f)(a, b) = \langle f, \psi_{a,b} \rangle = |a_0|^{-m/2} \int_{-\infty}^{+\infty} f(t) \overline{\psi(a_0^{-m}t - nb_0)} dt$$

特殊地，取  $a_0 = 2, b_0 = 1$ ，可以得到二进小波（dyadic wavelet）

$$\psi_{m,n}(t) = 2^{-m/2} \psi(2^{-m}t - n), m, n \in Z$$

在实际应用中，为了使小波变换的计算更加有效，通常构造的小波函数都具有正交性，即

$$\langle \psi_{m,n}, \psi_{j,k} \rangle = \int_{-\infty}^{+\infty} \psi_{m,n}(t) \overline{\psi_{j,k}(t)} dt = \delta_{m,j} \delta_{n,k}$$

从理论上可以证明，将连续小波变换离散成离散小波变换，信号的基本信息并不会丢失；相反，由于小波基函数的正交性，使得小波空间中两点之间因冗余度造成的关联得以消除。同时，正交性使得计算的误差更小，使得变换结果时-频函数更能反映信号本身的性质。

## 21.4 多分辨分析及 Mallat 算法

多分辨分析（Multi-Resolution Analysis, MRA）是由 S.Mallat 引入的。他从空间概念上形象地说明了小波的多分辨特性，将在此之前的所有小波变换理论统一起来。1989 年，Mallat 在小波变换多分辨分析理论与图像处理的应用研究中受到塔式算法的启发，提出了信号的塔式多分辨分析分解与重构的快速算法，即著名的 Mallat 算法。Mallat 算法在小波分析中的地位相当于快速傅里叶变换（FFT）在经典傅里叶分析中的地位。本节就介绍一维和二维张量积情况下的多分辨分析及相应的 Mallat 算法。至于三维及三维以上情况下的多分辨分析，由于在二维离散的数字图像的处理中不涉及，在此不做介绍。

## 21.5 一维正交多分辨分析及 Mallat 算法

多分辨分析是用小波函数的二进伸缩和平移表示函数这一思想的更加抽象复杂的表现形



式,它重点处理整个函数集,而非侧重处理作为个体的函数。MRA 形成了构造正交小波基的一个框架,常用的 B-样条正交小波基和 Daubechies 紧支撑正交小波基都可看作是该框架下的产物。下面就给出多分辨分析的严格定义。

定义 1 称  $L^2(R)$  中的闭子空间序列  $\{V_m\}_{m \in \mathbb{Z}}$  为一个 (二进) 多分辨分析, 如果  $\{V_k\}$  满足下列条件:

(1) 单调性:  $\dots \subset V_{j-1} \subset V_j \subset V_{j+1} \subset \dots, \forall j \in \mathbb{Z}$  ;

(2) 逼近性:  $\bigcap_{j \in \mathbb{Z}} V_j = \{0\}, \text{clos}_{L^2(R)}(\bigcup_{j \in \mathbb{Z}} V_j) = L^2(R)$  ;

(3) 伸缩性:  $f(t) \in V_j \Leftrightarrow f(2t) \in V_{j+1}, \forall j \in \mathbb{Z}$  ;

(4) 平移不变性:  $f(t) \in V_0 \Leftrightarrow f(t-k) \in V_0, \forall k \in \mathbb{Z}$  ;

(5) Riesz 基存在性: 存在函数  $\phi \in V_0$ , 使得  $\{\phi(t-k)\}_{k \in \mathbb{Z}}$  构成  $V_0$  的一个 Riesz 基, 即函数序列  $\{\phi(t-k)\}_{k \in \mathbb{Z}}$  线性无关, 且存在常数  $A$  和  $B$ , 满足  $0 < A \leq B < +\infty$ , 使得对任意的  $f(t) \in V_0$ , 总存在序列  $\{c_k\}_{k \in \mathbb{Z}} \in l^2$  使得

$$f(t) = \sum_{k=-\infty}^{+\infty} c_k \phi(t-k)$$

且  $A\|f\|_2^2 \leq \sum_{k=-\infty}^{+\infty} |c_k|^2 \leq B\|f\|_2^2$ , 则称  $\phi$  为尺度函数, 并称  $\phi$  生成  $L^2(R)$  的一个多分辨分析  $\{V_j\}_{j \in \mathbb{Z}}$ 。特别地, 若  $\{\phi(t-k)\}_{k \in \mathbb{Z}}$  构成  $V_0$  的一个标准正交基, 则称  $\phi$  为正交尺度函数; 相应地, 称  $\phi$  生成  $L^2(R)$  的一个正交多分辨分析  $\{V_j\}_{j \in \mathbb{Z}}$ 。

MRA 本质上给出了人类视觉系统对物体认识的数学描述。当人眼观察某一物体时, 如果设  $V_j$  为人眼在尺度  $j$  下所感知的物体信息 (如物体的一个侧面), 则当尺度增加到  $j+1$  时, 所摄取的信息量为  $V_{j+1}$  (如物体的全貌)。尺度的增加可以看作是观察距离的减小, 因而能更深入的认识物体, 即有  $V_j \subseteq V_{j+1}$ 。

在多分辨分析中, 称  $V_j$  为逼近空间。一般地, 不同的逼近空间  $V_j$  对应着不同的尺度函数, 从而对应  $L^2(R)$  不同的多分辨分析。在实际中, 比较有用的是一类具有紧支撑特性的尺度函数。

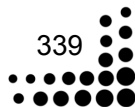
若  $\{V_j\}$  为尺度函数  $\phi$  生成的  $L^2(R)$  中的多分辨分析, 则必然存在系数序列  $\{h_k\}_{k \in \mathbb{Z}}$ , 使得以下尺度关系, 即两尺度方程成立:

$$\phi(t) = \sqrt{2} \sum_k h_k \phi(2t-k) \quad (21-4)$$

这是因为,  $\phi(t) \in V_0 \subset V_1$ , 且  $\{\sqrt{2}\phi(2t-k)\}_{k \in \mathbb{Z}}$  构成  $V_1$  的一个 Riesz 基, 故存在系数  $\{h_k\}$ , 使得  $\phi(t) = \sqrt{2} \sum_k h_k \phi(2t-k)$ 。尺度方程式 (21-4) 在频域的表达式为

$$\hat{\phi}(\omega) = 2^{-1/2} \hat{h}(\omega/2) \hat{\phi}(\omega/2) \quad (21-5)$$

其中,  $\hat{h}(\omega)$  为系数序列  $\{h_k\}_{k \in \mathbb{Z}}$  的傅里叶变换。假设  $\{h_k\}_{k \in \mathbb{Z}}$  是数字滤波器的脉冲响应, 则式 (21-5) 将离散滤波器和多尺度分析联系起来。





可以知道,若  $\phi(t) \in L^2(R)$ , 则  $\{\phi(t-n)\}_{n \in \mathbb{Z}}$  是标准(规范)正交系, 等价于恒等式

$$\sum_{k=-\infty}^{+\infty} |\hat{\phi}(\omega + 2k\pi)|^2 = 1$$

对于几乎所有的  $\omega \in R$  成立。

利用这个定理, 可以通过将尺度函数  $\phi$  正交化的方法, 得到一个正交尺度函数  $\phi$ 。若令

$$\hat{\phi}(\omega) = \frac{\hat{\phi}(\omega)}{\left[ \sum_{k \in \mathbb{Z}} |\hat{\phi}(\omega + 2k\pi)|^2 \right]^{1/2}} \quad (21-6)$$

则  $\phi \in V_0$ , 且  $\{\phi(t-k)\}_{k \in \mathbb{Z}}$  构成  $V_0$  的一个标准正交基, 而对于任意  $j, k \in \mathbb{Z}$ ,  $\phi_{j,k}(t) = 2^{j/2} \phi(2^j t - k)$  构成  $V_j$  一个标准正交基。从而  $\phi$  生成一个正交多分辨分析  $\{V_j\}$ 。

事实上, 对任意  $f(t) \in V_j$ , 由多分辨分析的定义可知,  $f(t/2^j) \in V_0$ 。则存在系数序列

$\{c_k\}_{k \in \mathbb{Z}} \in l^2$ , 使得  $f(t/2^j) = \sum_{k=-\infty}^{+\infty} c_k \phi(t-k)$ 。用  $2^j t$  代替  $t$ , 可得  $f(t) = \sum_{k=-\infty}^{+\infty} c_k \phi(2^j t - k)$ 。因此,

$\{\phi_{j,k}(t)\}_{k \in \mathbb{Z}}$  构成  $V_j$  的一个基。此外, 对于任意  $k, l \in \mathbb{Z}$ , 有

$$\langle \phi_{j,k}, \phi_{j,l} \rangle = 2^j \int_{-\infty}^{+\infty} \phi(2^j t - k) \phi^*(2^j t - l) dt = \int_{-\infty}^{+\infty} \phi(x - k) \phi^*(x - l) dx = \delta_{k,l}$$

则可以知道, 对于任意  $j \in \mathbb{Z}$ ,  $\{\phi_{j,k}(t)\}_{k \in \mathbb{Z}}$  是标准正交的。

但是, 通常情况下  $\{\phi_{j,k}(t)\}_{j,k \in \mathbb{Z}}$  不是  $L^2(R)$  的标准正交基。通过正交多分辨分析, 可以得到  $L^2(R)$  的一个标准正交基。

由于  $V_j \subset V_{j+1}$ , 则存在  $V_j$  在  $V_{j+1}$  中的正交补空间  $W_j$ , 使得

$$V_{j+1} = V_j \oplus W_j$$

上述空间正交分解过程可对逼近空间  $V_j$  递归进行下去, 可用图 21-1 表示。

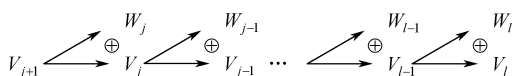


图 21-1 空间正交分解过程的递归

则  $V_{j+1} = V_l \oplus_{k=l}^j W_k$ ,  $l < j$ 。令  $j \rightarrow +\infty, l \rightarrow -\infty$ , 可得到  $L^2(R)$  的正交和分解

$$L^2(R) = \bigoplus_{k=-\infty}^{+\infty} W_k \quad (21-7)$$

如果  $\int_R \psi(t) dt = 0$ , 函数  $\psi(t) \in L^2(R)$  称为小波。如果  $\{\psi(t-k)\}_{k \in \mathbb{Z}}$  构成  $W_0$  的一个标准正交基, 则由  $V_{j+1} = V_j \oplus W_j$  可以推出  $\{\psi_{j,k} = 2^{j/2} \psi(2^j t - k)\}_{k \in \mathbb{Z}}$  构成  $W_j$  的标准正交基, 从而  $\{\psi_{j,k \in \mathbb{Z}}\}_{j,k \in \mathbb{Z}}$  构成  $L^2(R)$  的标准正交基, 这时称  $W_j$  为小波空间,  $\psi(t)$  为正交小波。

定理 1 设正交尺度函数  $\phi(t)$  生成了正交多分辨分析  $\{V_j\}_{j \in \mathbb{Z}}$ ,  $\{h_k\}_{k \in \mathbb{Z}}$  是满足两尺度方程的滤波器, 令



$$\psi(t) = \sqrt{2} \sum_n g_n \phi(2t - n) \quad (21-8)$$

其中,  $g_n$  满足  $g_k = (-1)^k h_{1-k}^*$ , 则  $\psi(t)$  为小波, 且它的平移系构成  $W_0$  的标准正交基, 其中  $W_0$  是  $V_0$  在  $V_1$  中的正交补, 从而  $\{\psi_{j,k}(t)\}_{j,k \in \mathbb{Z}}$  构成  $L^2(R)$  的一个标准正交基。

上述定理等价于以下几点:

- (1)  $\{\psi(t-k)\}_{k \in \mathbb{Z}}$  是标准正交的。
- (2) 对任意  $k \in \mathbb{Z}$ ,  $\psi(t-k) \in W_0$ 。
- (3)  $W_0$  中的每个函数都可表示为  $\psi(t-k)$  的线性组合。
- (4)  $\psi(t)$  是一个小波, 即  $\int_R \psi(t) dt = 0$ 。

定理 1 是一个非常重要的定理, 它给出了一种根据正交尺度函数来构造正交小波的一般方法。具体推导过程如下:

- (1) 利用式 (21-6), 由  $\hat{\phi}(\omega)$  计算  $\hat{\phi}(\omega)$ 。
- (2) 利用式 (21-5) 计算  $\hat{h}(\omega) = \sqrt{2}\hat{\phi}(2\omega)/\hat{\phi}(\omega)$ 。
- (3) 由  $g_k = (-1)^k h_{1-k}^*$ , 计算相应的傅里叶变换式  $\hat{g}(\omega) = -e^{-i\omega} \hat{h}^*(\omega + \pi)$ 。
- (4) 根据小波方程的频域表示式计算  $\hat{\psi}(\omega) = 2^{-1/2} \hat{g}(\omega/2) \hat{\phi}(\omega/2)$ 。

上述构造正交小波基的整个过程可总结为

$$\phi(t) \rightarrow \hat{\phi}(t) \rightarrow \hat{\phi}(\omega) \rightarrow \hat{h}(\omega) \rightarrow \hat{g}(\omega) \rightarrow \hat{\psi}(\omega) \rightarrow \psi(t)$$

根据上面的过程, 得到小波基之后, 就可以用其来表示  $L^2(R)$  中的任意函数  $f(t)$

$$f(t) = \sum_{j,k \in \mathbb{Z}} c_k^j \psi_{j,k}(t)$$

等式两边同时对  $\psi_{j,k}$  取内积, 并注意到  $\{\psi_{j,k}(t)\}_{j,k \in \mathbb{Z}}$  是  $L^2(R)$  的一个标准正交基, 可以得到  $c_k^j = \langle f, \psi_{j,k} \rangle$ , 则有

$$f(t) = \sum_{j,k \in \mathbb{Z}} \langle f, \psi_{j,k} \rangle \psi_{j,k}(t)$$

特别地, 对于  $L^2(R)$  中的任意子空间  $V_j$ , 有

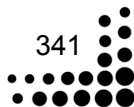
$$\begin{aligned} V_j &= V_{j-1} \oplus W_{j-1} \\ &= V_{j-2} \oplus W_{j-2} \oplus W_{j-1} = \dots \\ &= V_M \oplus W_M \oplus W_{M+1} \oplus \dots \oplus W_{j-1} (M < j) \end{aligned}$$

从而可以得到  $V_j$  中的任意函数  $f_j$  都存在如下多分辨分析表示:

$$\begin{aligned} f_j &= f_{j-1} \oplus d_{j-1} \\ &= f_{j-2} \oplus d_{j-2} \oplus d_{j-1} \\ &\dots \\ &= f_M \oplus d_M \oplus d_{M+1} \oplus \dots \oplus d_{j-1} (M < j) \end{aligned}$$

其中,

$$f_l(t) = \sum_k c_k^l \phi_{l,k} \in V_l, l = M, \dots, j$$





$$d_l(t) = \sum_k c_k^l \psi_{l,k} \in W_l, l = M, \dots, j-1$$

其中,  $f_M$  表示函数  $f_j$  的低频部分, 而  $d_l(t), l = M, \dots, j-1$  表示  $f_j$  在不同分辨率下的高频成分。

在下式

$$f_j(t) = \sum_k c_k^j \phi_{j,k}(t) = \sum_k c_k^{j-1} \phi_{j-1,k}(t) + \sum_k d_k^{j-1} \psi_{j,k}(t) \quad (21-9)$$

两边同时与  $\phi_{j-1,k}$  做内积, 并利用  $\phi$   $\psi$  及其二进伸缩和平移的正交特性, 可得到

$$c_k^{j-1} = \sum_n c_n^j \langle \phi_{j,k}(t), \phi_{j-1,k} \rangle = \sum_n c_n^j h_{n-2k}^*$$

类似地, 有

$$d_k^{j-1} = \sum_n c_n^j \langle \phi_{j,k}(t), \psi_{j-1,k} \rangle = \sum_n c_n^j g_{n-2k}^*$$

同时对式 (21-9) 两边与  $\phi_{j,k}$  做内积, 有

$$\begin{aligned} c_k^j &= \sum_n c_n^{j-1} \langle \phi_{j,k}(t), \phi_{j,k} \rangle + \sum_n d_n^{j-1} \langle \psi_{j,k}(t), \phi_{j,k} \rangle \\ &= \sum_n c_n^{j-1} h_{k-2n} + \sum_n d_n^{j-1} g_{k-2n} \end{aligned}$$

其中,  $\{h_k\}_{k \in \mathbb{Z}}$  是由正交尺度函数的两尺度方程对应的滤波器系数序列, 可看作是低通滤波器;

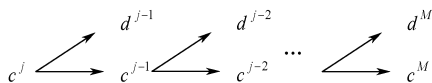
$\{g_k\}_{k \in \mathbb{Z}}$  由式  $g_k = (-1)^k h_{1-k}^*$  给出, 可看作是高通滤波器。由此, 可以得到一维情况下离散小波

变换的 Mallat 算法。其卷积表达形式为

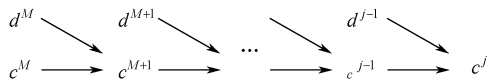
$$\begin{cases} c^{j-1} = D(c^j * \bar{h}^*) \\ d^{j-1} = D(c^j * \bar{g}^*) \end{cases}$$

$$c^j = (Uc^{j-1}) * h + (Ud^{j-1}) * g$$

其中,  $\bar{h}^*$  表示滤波器  $h$  的共轭反转;  $c^j * \bar{h}^*$  表示  $c^j$  与  $\bar{h}^*$  的卷积;  $D(c^j * \bar{h}^*)$  表示对卷积  $c^j * \bar{h}^*$  的二元下抽样;  $Uc^{j-1}$  表示对序列  $c^{j-1}$  的二元上抽样;  $U$ 、 $D$  分别为二元上、下抽样算子。小波分解与重构的迭代过程如图 21-2 所示。相应的二通道滤波器组表示如图 21-3 所示。



(a) 小波分解过程



(b) 小波重构过程

图 21-2 一维信号小波分解与重构的迭代过程

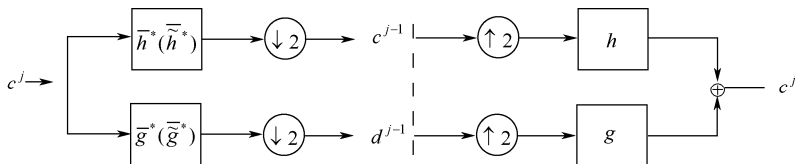


图 21-3 一维信号小波分解与重构的二通道滤波器组表示 (括号中选项表示双正交滤波器)



由于除了 Haar 小波外, 没有同时具有对称性和紧支撑性的正交小波。紧支撑性, 意味着小波滤波器的长度是有限的, 以便于计算 (如果长度无限, 将难以处理)。对称性, 意味着小波滤波器的线性相位, 线性相位可以保持原信号的相位不变。紧支撑性和对称性在信号处理中具有重要的意义。为了同时具有这两种性质, 就必须放弃小波的正交性, 而采用双正交小波。下面就来介绍双正交多分辨分析的概念、双正交小波基在函数多分辨表示中的作用及双正交小波分解与重构的滤波器组算法。

定义 2 设函数  $\psi(t) \in L^2(R)$ , 称  $\{\psi_{j,k}(t)\}_{j,k \in \mathbb{Z}}$  是  $L^2(R)$  的一个 Riesz 基, 如果它是线性无关的, 且存在常数  $A$  和  $B$ , 满足  $0 < A \leq B < +\infty$ , 则对任意  $f(t) \in L^2(R)$ , 总存在序列  $\{c_{j,k}\}_{j,k \in \mathbb{Z}} \in l^2(R^2)$  满足

$$f(t) = \sum_{j=-\infty}^{+\infty} \sum_{k=-\infty}^{+\infty} c_{j,k} \psi_{j,k}(t)$$

$$\text{且 } A\|f\|_2^2 \leq \sum_{k=-\infty}^{+\infty} |c_{j,k}|^2 \leq B\|f\|_2^2.$$

设  $\{V_j\}_{j \in \mathbb{Z}}$  和  $\{\tilde{V}_j\}_{j \in \mathbb{Z}}$  是  $L^2(R)$  的两个多分辨分析, 它们的尺度函数分别为  $\phi$  和  $\tilde{\phi}$ , 如果  $\phi$  和  $\tilde{\phi}$  满足如下正交性条件:

$$\langle \phi(\bullet - j), \tilde{\phi}(\bullet - k) \rangle = \delta_{j,k}, \forall j, k \in \mathbb{Z} \quad (21-10)$$

则称  $\phi$  和  $\tilde{\phi}$  为对偶尺度函数。

令  $W_j$  是  $V_j$  在  $V_{j+1}$  中的补空间,  $\tilde{W}_j$  是  $\tilde{V}_j$  在  $\tilde{V}_{j+1}$  中的补空间, 使得

$$V_{j+1} = V_j \dot{+} W_j, \tilde{V}_{j+1} = \tilde{V}_j \dot{+} \tilde{W}_j$$

其中,  $\dot{+}$  为直和运算。

如果存在  $L^2(R)$  中的函数  $\psi(t)$ ,  $\tilde{\psi}(t)$  和序列  $\{g_n\}$ ,  $\{\tilde{g}_n\}$  满足

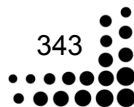
$$\begin{cases} \psi(t) = \sqrt{2} \sum_n g_n \phi(2t - n) \\ \tilde{\psi}(t) = \sqrt{2} \sum_n \tilde{g}_n \tilde{\phi}(2t - n) \end{cases} \quad (21-11)$$

使得

$$\begin{cases} \langle \psi_{j,k}, \tilde{\psi}_{l,m} \rangle = \delta_{j,l} \delta_{k,m}, \forall j, k, l, m \in \mathbb{Z} \\ \langle \psi_{j,k}, \tilde{\phi}_{j,m} \rangle = 0, \forall j, k, m \in \mathbb{Z} \\ \langle \phi_{j,k}, \tilde{\psi}_{j,m} \rangle = 0, \forall j, k, m \in \mathbb{Z} \end{cases} \quad (21-12)$$

且  $\{\psi_{j,k}(t)\}_{k \in \mathbb{Z}}$  是  $W_j$  的一个 Riesz 基,  $\{\tilde{\psi}_{j,k}(t)\}_{k \in \mathbb{Z}}$  是  $\tilde{W}_j$  的一个 Riesz 基, 从而  $\{\psi_{j,k}(t)\}_{j,k \in \mathbb{Z}}$  和  $\{\tilde{\psi}_{j,k}(t)\}_{j,k \in \mathbb{Z}}$  都构成  $L^2(R)$  的 Riesz 基, 则称  $\{V_j, \tilde{V}_j\}_{j \in \mathbb{Z}}$  是  $L^2(R)$  的一个双正交多分辨分析。这时, 称  $\psi(t)$  和  $\tilde{\psi}(t)$  为对偶双正交小波。

在双正交多分辨分析的定义中, 如果  $\tilde{\phi} = \phi, \tilde{\psi} = \psi$ , 则由式 (21-10) 可知,  $\phi$  为正交尺度函数, 它生成  $L^2(R)$  的一个正交多分辨分析  $\{V_j\}_{j \in \mathbb{Z}}$ 。这说明正交多分辨分析是双正交多分辨







分析的特殊情况。类似正交小波分解和重构过程的推导，可以得到双正交小波变换的 Mallat 算法如下：

$$\begin{cases} c^{j-1} = D(c^j * \tilde{h}^*) \\ d^{j-1} = D(c^j * \tilde{g}^*) \\ c^j = (Uc^{j-1}) * h + (Ud^{j-1}) * g \end{cases}$$

选取图 21-3 中括号中的滤波器，可以得到双正交小波分解与重构对应的二通道滤波器组表示。

## 21.6 紧支撑双正交小波基的构造

由于除了 Haar 小波外，没有同时具有对称性和紧支撑性的正交小波，而 Haar 小波不能很好地表示和分析连续信号或函数，因而在数字图像处理中，常常采用同时具有对称性和紧支撑性的双正交小波。故本节只介绍具有紧支撑性的双正交小波基的构造方法。正交小波基的构造方法可见参考文献相关部分的介绍。

设  $\tilde{\phi}$ 、 $\phi$  是对偶的双正交尺度函数， $\tilde{\psi}$ 、 $\psi$  是对偶双正交小波，则有与其相应的两尺度方程和小波方程

$$\begin{cases} \phi(t) = \sqrt{2} \sum_k h_k \phi(2t - k) \\ \tilde{\phi}(t) = \sqrt{2} \sum_k \tilde{h}_k \tilde{\phi}(2t - k) \end{cases} \quad (21-13)$$

$$\begin{cases} \psi(t) = \sqrt{2} \sum_k g_k \phi(2t - k) \\ \tilde{\psi}(t) = \sqrt{2} \sum_k \tilde{g}_k \tilde{\phi}(2t - k) \end{cases} \quad (21-14)$$

将式 (21-13) 和式 (21-14) 中的每个方程两边都同时积分，可得

$$\begin{cases} \sum_k h_k = \sqrt{2} \\ \sum_k \tilde{h}_k = \sqrt{2} \end{cases} \quad (21-15)$$

$$\begin{cases} \sum_k g_k = 0 \\ \sum_k \tilde{g}_k = 0 \end{cases} \quad (21-16)$$

容易推出，式 (21-13) 和式 (21-14) 等价的频域表示形式分别为

$$\begin{cases} \hat{\phi}(\omega) = 2^{-1/2} \hat{h}(\omega/2) \hat{\phi}(\omega/2) \\ \hat{\tilde{\phi}}(\omega) = 2^{-1/2} \hat{\tilde{h}}(\omega/2) \hat{\tilde{\phi}}(\omega/2) \end{cases} \quad (21-17)$$

$$\begin{cases} \hat{\psi}(\omega) = 2^{-1/2} \hat{g}(\omega/2) \hat{\phi}(\omega/2) \\ \hat{\tilde{\psi}}(\omega) = 2^{-1/2} \hat{\tilde{g}}(\omega/2) \hat{\tilde{\phi}}(\omega/2) \end{cases} \quad (21-18)$$



对式 (21-17) 和式 (21-18) 中的各方程进行反复迭代, 可得各式的无穷积表示

$$\hat{\phi}(\omega) = \prod_{j=1}^{\infty} \frac{\hat{h}(\omega/2^j)}{\sqrt{2}}, \quad \hat{\phi}(\omega) = \prod_{j=1}^{\infty} \frac{\hat{h}(\omega/2^j)}{\sqrt{2}} \quad (21-19)$$

$$\hat{\psi}(\omega) = \prod_{j=1}^{\infty} \frac{\hat{g}(\omega/2^j)}{\sqrt{2}}, \quad \hat{\psi}(\omega) = \prod_{j=1}^{\infty} \frac{\hat{g}(\omega/2^j)}{\sqrt{2}} \quad (21-20)$$

这说明式 (2-19) 和式 (2-20) 在  $L^2(R)$  中都是收敛的。

另外, 滤波器组  $h, \tilde{h}, g, \tilde{g}$  还要满足完全重构条件

$$\begin{cases} \sum_k h_k \tilde{h}_{k+2n} = \delta_{0,n} \\ \tilde{g}_n = (-1)^n h_{1-n} \\ g_n = (-1)^n \tilde{h}_{1-n} \end{cases} \quad (21-21)$$

由式 (21-15) 式 (21-16) 和式 (21-21) 容易推出

$$\sum_k h_{2k} = \sum_k h_{2k+1} = 1/\sqrt{2}, \quad \sum_k \tilde{h}_{2k} = \sum_k \tilde{h}_{2k+1} = 1/\sqrt{2}$$

此外, 由式 (21-15) 和式 (21-21) 可以推出式 (21-16); 由式 (21-19) 和式 (21-13) 可以推出式 (21-20)。

综上可以得到, 有限滤波器  $h, \tilde{h}, g, \tilde{g}$  使得尺度函数  $\tilde{\phi}, \phi$  和对偶小波函数  $\tilde{\psi}, \psi$  存在的必要条件为

$$\begin{cases} \sum_k h_k \tilde{h}_{k+2n} = \delta_{0,n} \\ \sum_k h_{2k} = \sum_k h_{2k+1} = 1/\sqrt{2} \\ \sum_k \tilde{h}_{2k} = \sum_k \tilde{h}_{2k+1} = 1/\sqrt{2} \end{cases} \quad (21-22)$$

及

$$\begin{cases} \tilde{g}_n = (-1)^n h_{1-n} \\ g_n = (-1)^n \tilde{h}_{1-n} \end{cases} \quad (21-23)$$

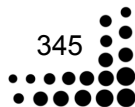
式 (21-22) 称为双正交滤波器的约束条件。

下面的定理说明, 对完全重构的有限长滤波器  $h, \tilde{h}, g, \tilde{g}$  给出了生成  $L^2(R)$  的一个双正交基的充分条件。

定理 2 若有限长滤波器  $h, \tilde{h}, g, \tilde{g}$  满足式 (21-21),  $h(\omega), \hat{h}(\omega)$  满足

$$\begin{cases} \hat{h}(\omega) = \sqrt{2} \left( \frac{1 + e^{-i\omega}}{2} \right)^p F_0(e^{i\omega}) \\ \hat{\tilde{h}}(\omega) = \sqrt{2} \left( \frac{1 + e^{-i\omega}}{2} \right)^{\tilde{p}} \tilde{F}_0(e^{i\omega}) \end{cases}$$

其中,  $\omega = \pi$  时,  $F_0(e^{i\omega}) \neq 0, \tilde{F}_0(e^{i\omega}) \neq 0$ , 并且





$$\begin{cases} \max_{\omega \in R} \left| \prod_{i=1}^{p-1} F_0(2^i \omega) \right| < 2^p \\ \max_{\omega \in R} \left| \prod_{i=1}^{\tilde{p}-1} \tilde{F}_0(2^i \omega) \right| < 2^{\tilde{p}} \end{cases}$$

则

(1) 无穷积  $\prod_{j=1}^{\infty} \frac{\hat{h}(\omega/2^j)}{\sqrt{2}}$ 、 $\prod_{j=1}^{\infty} \frac{\hat{\tilde{h}}(\omega/2^j)}{\sqrt{2}}$  分别收敛于  $L^2(R)$  中的函数  $\hat{\phi}(\omega)$ 、 $\hat{\tilde{\phi}}(\omega)$ 。

(2)  $\phi$ 、 $\tilde{\phi}$  满足双正交关系。

(3) 两个小波族  $\{\psi_{j,n}\}_{j,n \in \mathbb{Z}}$ 、 $\{\tilde{\psi}_{j,n}\}_{j,n \in \mathbb{Z}}$  都是  $L^2(R)$  的双正交 Riesz 基, 其中  $\psi$ 、 $\tilde{\psi}$  满足式 (21-13)。且相应的尺度函数  $\phi$ 、 $\tilde{\phi}$  和小波函数  $\psi$ 、 $\tilde{\psi}$  也都有紧支集。具体地, 如果对于  $N_1 \leq n \leq N_2$ ,  $h_n \neq 0$ ; 对于  $\tilde{N}_1 \leq \tilde{n} \leq \tilde{N}_2$ ,  $\tilde{h}_{\tilde{n}} \neq 0$ , 使得  $h$ 、 $\tilde{h}$  的支撑分别为  $[N_1, N_2]$  和  $[\tilde{N}_1, \tilde{N}_2]$ , 则  $\phi$ 、 $\tilde{\phi}$  的支撑分别为  $[N_1, N_2]$  和  $[\tilde{N}_1, \tilde{N}_2]$ 。因而, 相应的小波  $\psi$ 、 $\tilde{\psi}$  的支撑分别为  $\left[\frac{N_1 - \tilde{N}_2 + 1}{2}, \frac{N_2 - \tilde{N}_1 + 1}{2}\right]$  和  $\left[\frac{\tilde{N}_1 - N_2 + 1}{2}, \frac{\tilde{N}_2 - N_1 + 1}{2}\right]$ 。所以两个小波的支撑长度相同, 都等于  $(N_2 - N_1 + \tilde{N}_2 - \tilde{N}_1)/2$ 。

设  $h = \{h_n\}$  和  $\tilde{h} = \{\tilde{h}_n\}$  是关于 0 对称的对偶低通滤波器, 即满足

$$\begin{cases} h_n = h_{-n} \\ \tilde{h}_n = \tilde{h}_{-n} \end{cases} \quad (21-24)$$

且  $h$ 、 $\tilde{h}$  的长度分别为  $2N+1$  和  $2\tilde{N}+1$ , 则有下面的结论成立。

(1) 小波  $\psi$  有  $\tilde{p}$  阶消失矩等价于  $h$  满足如下关系

$$\begin{cases} h_0 - 2h_1 + 2h_2 - 2h_3 + \cdots + 2(-1)^N h_N = 0 \\ (-1)h_1 + (-1)^2 2^k h_2 + \cdots + (-1)^N N^k h_N = 0, 0 < k < \tilde{p} \text{ 且 } k \text{ 为偶数} \end{cases} \quad (21-25)$$

(2) 小波  $\tilde{\psi}$  有  $p$  阶消失矩等价于  $\tilde{h}$  满足如下关系:

$$\begin{cases} \tilde{h}_0 - 2\tilde{h}_1 + 2\tilde{h}_2 - 2\tilde{h}_3 + \cdots + 2(-1)^{\tilde{N}} \tilde{h}_{\tilde{N}} = 0 \\ (-1)\tilde{h}_1 + (-1)^2 2^k \tilde{h}_2 + \cdots + (-1)^{\tilde{N}} \tilde{N}^k \tilde{h}_{\tilde{N}} = 0, 0 < k < p \text{ 且 } k \text{ 为偶数} \end{cases} \quad (21-26)$$

综上所述可知, 构造具有  $\tilde{p}$ 、 $p$  阶消失矩并关于 0 对称的双正交小波  $\psi$  和  $\tilde{\psi}$  的约束条件为式 (21-22)、式 (21-24)、式 (21-26)。

下面举例说明双正交小波基的构造方法。

**【例 21-1】**构造(5-3)小波滤波器。该滤波器被广泛应用于图像压缩, 而且为 JPEG 2000 中无损图像压缩的默认滤波器。 $\tilde{h}$ 、 $h$  的滤波器长度分别为 5 和 3, 即  $\tilde{N}=2$ 、 $N=1$ ,  $\tilde{p}=2$ 、 $p=2$ , 并且是关于 0 对称的双正交小波。可以得到小波滤波器的约束条件为

$$\begin{cases} h_0\tilde{h}_0 + 2h_1\tilde{h}_1 = 1 \\ h_0\tilde{h}_2 + h_1\tilde{h}_1 = 0 \\ h_0 = 1/\sqrt{2} \\ 2h_1 = 1/\sqrt{2} \\ \tilde{h}_0 + 2\tilde{h}_2 = 1/\sqrt{2} \\ 2\tilde{h}_1 = 1/\sqrt{2} \end{cases}$$

整理并求解该方程组可得到

$$\tilde{h} = \{-1/4\sqrt{2}, 1/2\sqrt{2}, 3/2\sqrt{2}, 1/2\sqrt{2}, -1/4\sqrt{2}\}, h = \{1/2\sqrt{2}, 1/\sqrt{2}, 1/2\sqrt{2}\}$$

【例 21-2】构造 (9-7) 小波滤波器。该滤波器被 JPEG 2000 用于有损图像压缩的默认滤波器。

$\tilde{h}$ 、 $h$  的滤波器长度分别为 9 和 7，即  $\tilde{N}=4$ 、 $N=3$ ， $\tilde{p}=4$ 、 $p=4$ ，并且是关于 0 对称的双正交小波。可以得到 (9-7) 小波滤波器的约束条件为

$$\begin{cases} h_2\tilde{h}_4 + h_3\tilde{h}_3 = 0 \\ h_0\tilde{h}_4 + h_1\tilde{h}_3 + h_2\tilde{h}_2 + h_3\tilde{h}_1 = 0 \\ h_2\tilde{h}_4 + h_1\tilde{h}_3 + h_0\tilde{h}_2 + h_1\tilde{h}_1 + h_2\tilde{h}_0 + h_3\tilde{h}_1 = 0 \\ 2h_3\tilde{h}_3 + 2h_2\tilde{h}_2 + 2h_1\tilde{h}_1 + h_0\tilde{h}_0 = 1 \\ h_0 + 2h_2 = 1/\sqrt{2} \\ h_1 + h_3 = 1/2\sqrt{2} \\ \tilde{h}_0 + 2\tilde{h}_2 + 2\tilde{h}_4 = 1/\sqrt{2} \\ \tilde{h}_1 + \tilde{h}_3 = 1/2\sqrt{2} \\ -h_1 + 4h_2 - 9h_3 = 0 \\ -\tilde{h}_1 + 4\tilde{h}_2 - 9\tilde{h}_3 + 16\tilde{h}_4 = 0 \end{cases}$$

整理并求解该方程组可得到  $\tilde{h}$ 、 $h$  的值，见表 21-1。

表 21-1 小波与对偶小波都具有 4 阶消失矩的(9-7)小波滤波器系数

$p, \tilde{p}$	$n$	$h_n$	$\tilde{h}_n$
$p=4$ $\tilde{p}=4$	0	0.78848561640558	0.85269867900889
	1, -1	0.41809227322162	0.37740285561283
	2, -2	-0.04068941760916	-0.11062440441844
	3, -3	-0.06453888262870	-0.02384946501956
	4, -4	0	0.03782845550726

## 21.7 第二代小波变换

由于一般的小波滤波器的输出结果是浮点数，因而在对变换后的数据进行压缩时，要先进行量化，以得到相应的整数，这必然会引入误差，不适合于图像的无损压缩。



1994 年, Wim Swelden 提出了一种新的小波构造方法——提升方案 (Lifting Scheme), 也叫第二代小波变换 (Second Generation Wavelet Transform, SGWT) 或[整数到]整数小波变换 ([Integer-to-]Integer Wavelet Transform, [IT]IWT)。

第二代小波变换构造方法的特点是:

- (1) 继承了第一代小波的多分辨率的特性。
- (2) 不依赖傅里叶变换, 直接在时域完成小波变换。
- (3) 小波变换后的系数可以是整数。
- (4) 图像的恢复质量与变换时边界采用何种延拓方式无关。

第二代小波变换是由第一代小波变换的提升实现的。与第一代小波相比, 第二代小波还具有以下优点:

- (1) 算法简单、速度快、适合并行处理。
- (2) 对内存的需求量小, 便于 DSP 芯片实现。
- (3) 可用本位操作进行运算, 能实现任意图像尺寸的小波变换。

由于第二代小波能实现图像的整数到整数的变换, 因此给图像的无损压缩提供了理论基础, 是 JPEG 2000 标准的一个组成部分。

小波提升是一种构造紧支集双正交小波的新方法。

由提升构成第二代小波变换的过程分为如下 3 个步骤。

#### (1) 分裂。

分裂 (Split) 是将原始信号  $s_j = \{s_j, k\}$  分为两个互不相交的子集和。每个子集的长度是原子集的一半。通常是将一个数列分为偶数序列  $e_{j-1}$  和奇数序列  $o_{j-1}$ , 即

$$\text{Split}(s_j) = (e_{j-1}, o_{j-1})$$

其中,  $e_{j-1} = \{e_{j-1}, k = s_{j,2k}\}$ ,  $o_{j-1} = \{o_{j-1}, k = s_{j,2k+1}\}$ 。

#### (2) 预测。

预测 (Predict) 是利用偶数序列和奇数序列之间的相关性, 由其中一个序列 (一般是偶数序列  $e_{j-1}$ ) 来预测另一个序列 (一般是奇数序列  $o_{j-1}$ )。实际值  $o_{j-1}$  与预测值  $P(e_{j-1})$  的差值  $d_{j-1}$  反映了两者的逼近程度, 称为细节系数或小波系数, 对应于原信号  $s_j$  的高频部分。一般来说, 数据的相关性越强, 则小波系数的幅值就越小。如果预测是合理的, 则差值数据集  $d_{j-1}$  所包含的信息比原始子集  $o_{j-1}$  包含的信息要少得多。预测过程如下:

$$d_{j-1} = o_{j-1} - P(e_{j-1})$$

其中, 预测算子  $P$  可用预测函数  $P_k$  来表示, 函数  $P_k$  可取为  $e_{j-1}$  中的对应数据本身, 即

$$P_k(e_{j-1}, k) = e_{j-1}, k = s_{j,2k}$$

或  $e_{j-1}$  中的对应数据的相邻数据的平均值

$$P_k(e_{j-1}) = (e_{j-1,k} + e_{j-1,k+1})/2 = (s_{j,2k} + s_{j,2k+1})/2$$

或其他更复杂的函数。

#### (3) 更新。

经过分裂步骤产生子集的某些整体特征(如均值)可能与原始数据并不一致, 为了保持原始数据的这些整体特征, 需要一个更新 (Update) 过程。将更新过程用算子  $U$  来代替, 其过程如下:

$$s_{j-1} = e_{j-1} + U(d_{j-1})$$



其中,  $s_{j-1}$  为  $s_j$  的低频部分; 与预测函数一样, 更新算子也可以取不同函数, 如

$$U_k(d_{j-1})=d_{j-1,k}/2$$

或  $U_k(d_{j-1})=(d_{j-1,k-1}+d_{j-1,k})/4+1/2$ 。

$P$  与  $U$  取不同的函数, 可构造出不同的小波变换。

经过小波提升, 可将信号  $s_j$  分解为低频部分  $s_{j-1}$  和低频部分  $d_{j-1}$ ; 对于低频数据子集  $s_{j-1}$  可以再进行相同的分裂、预测和更新, 把  $s_{j-1}$  进一步分解成  $d_{j-2}$  和  $s_{j-2}$ ……如此下去, 经过  $n$  次分解后, 原始数据  $s_j$  的小波表示为  $\{s_{j-n}, d_{j-n}, d_{j-n+1}, \dots, d_{j-1}\}$ 。其中,  $s_{j-n}$  代表了信号的低频部分, 而  $\{d_{j-n}, d_{j-n+1}, \dots, d_{j-1}\}$  则是信号的从低到高的低频部分系列。

每次分解都对应于上面的 3 个提升步骤——分裂、预测和更新, 即

$$\text{Split}(s_j)=(e_{j-1}, o_{j-1}), d_{j-1}=o_{j-1}-P(e_{j-1}), s_{j-1}=e_{j-1}+U(d_{j-1})$$

小波提升是一个完全可逆的过程, 其反变换的步骤如下:

$$e_{j-1}=s_{j-1}-U(d_{j-1}), o_{j-1}=d_{j-1}+P(e_{j-1}), s_j=\text{Merge}(e_{j-1}, o_{j-1})$$

如图 21-4 所示是用提升方法进行小波分解和重构的示意图。

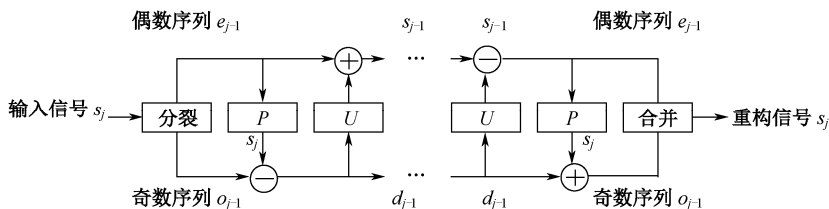


图 21-4 提升方法分解和重构

分解的 3 个步骤可以用替代的方式来计算: 先将奇数序列更新 (用偶数序列预测奇数序列), 然后用更新的奇数序列更新偶数序列。大致过程如下:

$$\text{Split}(s_j)=(e_{j-1}, o_{j-1}), o_{j-1}+=P(e_{j-1}), e_{j-1}+=U(o_{j-1})$$

其反变换过程也可以用替代的方式来计算:

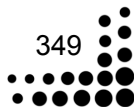
$$e_{j-1}+=U(o_{j-1}), o_{j-1}+=P(e_{j-1}), s_j=\text{Merge}(e_{j-1}, o_{j-1})$$

由上面的讨论可见, 基于上升型模式的第二代小波变换具有如下特点:

- (1) 本位操作: 所有运算可做本位操作, 节省内存。
- (2) 效率高: 利用复合赋值, 减少了浮点运算量。
- (3) 并行性: 一个上升步骤中的所有操作是并行的, 而多个上升步骤之间是串行的。
- (4) 逆变换: 逆变换只需简单地改变代码执行的先后循序, 具有与正向变换相同的计算复杂性。
- (5) 通用性: 变换过程中不必依赖 Fourier 分析, 很容易推广到一般性应用领域。
- (6) 非线性: 易于构造非线性小波变换 (如整数变换)。
- (7) 自适应: 支持自适应性小波变换。函数的分析由粗到细逐步进行, 细化过程可仅限于感兴趣的区域。

第二代小波变换实现方式相对于第一代小波变换具有如下优势: 在进行提升计算时, 可以采用替代的方法, 因此能节省大量空间; 通过子表达式的重复使用, 需要计算轮廓和细节部分的浮点操作数目大大减少, 因此能提高效率。

Haar 小波最初是由数学家 Haar 在 1910 年提出的, 那时还没有小波的概念, 叫作 Haar





函数。按现在的观点，它属于正交小波。下面讨论线性 Haar 小波变换。

### (1) 线性 Haar 小波变换。

取预测函数

$$P_k(e_{j-1})=e_{j-1,k}=s_{j,2k}$$

更新函数

$$U_k(d_{j-1})=d_{j-1,k}/2$$

则得到线性 Haar 小波变换。

分解式如下：

$$\begin{aligned} \text{Split}(s_j) &= (e_{j-1}, o_{j-1}) \\ d_{j-1,k} &= o_{j-1,k} - P_k(e_{j-1}) = o_{j-1,k} - e_{j-1,k} = s_{j,2k+1} - s_{j,2k} \\ s_{j-1,k} &= e_{j-1,k} + U_k(d_{j-1}) = s_{j,2k} + d_{j-1,k}/2 = (s_{j,2k+1} + s_{j,2k})/2 \end{aligned}$$

重构式如下：

$$\begin{aligned} e_{j-1,k} &= s_{j-1,k} - U_k(d_{j-1}) = s_{j-1,k} - d_{j-1,k}/2 \\ o_{j-1,k} &= d_{j-1,k} + P_k(e_{j-1}) = d_{j-1,k} + e_{j-1,k} \\ s_j &= \text{Merge}(e_{j-1}, o_{j-1}) \end{aligned}$$

### (2) 线性小波变换。

取预测函数

$$P_k(e_{j-1}) = (e_{j-1,k} + e_{j-1,k+1})/2 = (s_{j,2k} + s_{j,2k+2})/2$$

更新函数

$$U_k(d_{j-1}) = (d_{j-1,k-1} + d_{j-1,k})/4$$

则得到线性小波变换，如图 21-5 所示。

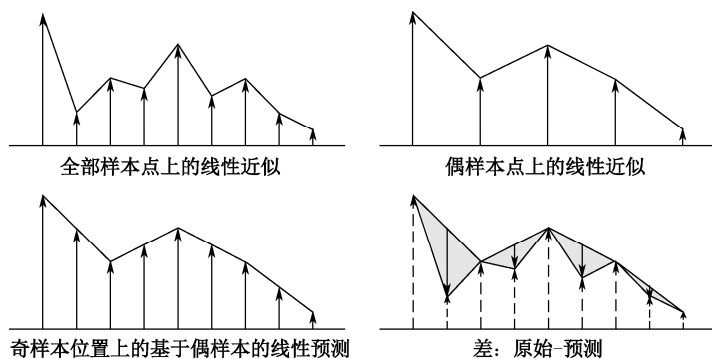


图 21-5 线性小波变换

分解式如下：

$$\begin{aligned} \text{Split}(s_j) &= (e_{j-1}, o_{j-1}) \\ d_{j-1,k} &= o_{j-1,k} - P_k(e_{j-1}) = o_{j-1,k} - (e_{j-1,k} + e_{j-1,k+1})/2 = s_{j,2k+1} - (s_{j,2k} + s_{j,2k+2})/2 \\ s_{j-1,k} &= e_{j-1,k} + U_k(d_{j-1}) = s_{j,2k} + (d_{j-1,k-1} + d_{j-1,k})/4 \end{aligned}$$

重构式如下：

$$\begin{aligned} e_{j-1,k} &= s_{j-1,k} - U_k(d_{j-1}) = s_{j-1,k} - (d_{j-1,k-1} + d_{j-1,k})/4 \\ o_{j-1,k} &= d_{j-1,k} + P_k(e_{j-1}) = d_{j-1,k} + (e_{j-1,k} + e_{j-1,k+1})/2 \end{aligned}$$



$$s_j = \text{Merge}(e_{j-1}, o_{j-1})$$

实际上，提升算法是一种改善快速小波变换的方法，单步的提升算法并不能用于所有的小波构造过程。事实上，只有一些特殊的小波变换很容易用它构造，如双正交小波。不过，涉及有限滤波器（FIR）的所有小波或子带变换可用多个提升步骤来构造。Daubechies 和 Sweldens 等已经证明，借助于因子化小波变换，所有小波的构造都能够用提升模式实现。

### （3）整数小波变换。

可以用提升方法来构造具紧支集的双正交小波，那么就可以通过对每次滤波后的数据进行取整（用  $[\cdot]$  表示）来实现整数小波变换，而且这种变换是完全可逆的，也就是完全重构数据。

Sweldens 已经证明在提升的基础上可以进行整数集到整数集的小波变换，也就是说，一个整数集合通过小波变换得到的仍然是整数集合。这就给数字图像的压缩编码带来了好处，由于不需要对变换后的系数进行量化，因此提供了实现无损压缩的可能。

下面是几个典型的整数小波变换的例子。

#### （1）S 变换。

最简单的整数小波变换是 S 变换（Stransform, S=Sequential），它是线性 Haar 小波变换的近似整数形式。分解式如下：

$$\begin{aligned} d_{j-1,k} &= s_{j,2k+1} - s_{j,2k} \\ s_{j-1,k} &= s_{j,2k} + [d_{j-1,k}/2] \end{aligned}$$

相当于对原更新函数取整。

重构式如下：

$$\begin{aligned} s_{j,2k} &= s_{j-1,k} - [d_{j-1,k}/2] \\ s_{j,2k+1} &= d_{j-1,k} + s_{j,2k} \end{aligned}$$

#### （2）S+P 变换。

S 变换之后，在低通系数  $s_{j-1,k}$  的基础上进行线性预测，以产生新的高通系数  $d_{j-1,k}$ ，这就是 S+P 变换族（S+P family of transform, S+P=Sequential Plus Prediction）。分解式如下：

$$\begin{aligned} s_{j-1,k} &= s_{j,2k} + [v_k/2] \\ d_{j-1,k} &= v_k + [t_k + 1/2] \end{aligned}$$

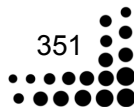
其中，

$$\begin{aligned} v_k &= s_{j,2k+1} - s_{j,2k} \\ t_k &= \alpha_{-1}(s_{j-1,k-2} - s_{j-1,k-1}) + \alpha_0(s_{j-1,k-1} - s_{j-1,k}) + \alpha_1(s_{j-1,k} - s_{j-1,k+1}) + \beta_{-1}v_{k+1} \end{aligned}$$

例如，取参数如表 21-2 所示。

表 21-2 S+P 变换的参数表

变换	$\alpha_{-1}$	$\alpha_0$	$\alpha_1$	$\beta_{-1}$
S	0	0	0	0
2/6	0	1/4	1/4	0
B	0	1/4	3/8	1/4
C	-1/16	1/4	1/2	3/8







其中

S 变换。

$$s_{j-1,k} = s_{j,2k} + [v_k/2] = s_{j,2k} + [(s_{j,2k+1} - s_{j,2k})/2]$$

$$d_{j-1,k} = v_k = s_{j,2k+1} - s_{j,2k}$$

其分解与重构式同上式。

2/6 变换。

分解：

$$s_{j-1,k} = s_{j,2k} + [v_k/2] = s_{j,2k} + [(s_{j,2k+1} - s_{j,2k})/2]$$

$$d_{j-1,k} = v_k + [(s_{j-1,k-1} - s_{j-1,k})/4 + (s_{j-1,k} - s_{j-1,k+1})/4 + 1/2]$$

$$= (s_{j,2k+1} - s_{j,2k}) + [(s_{j-1,k-1} - s_{j-1,k+1})/4 + 1/2]$$

即

$$v_k = s_{j,2k+1} - s_{j,2k}$$

$$s_{j-1,k} = s_{j,2k} + [v_k/2]$$

$$d_{j-1,k} = v_k + [(s_{j-1,k-1} - s_{j-1,k+1})/4 + 1/2]$$

重构：

$$u_k = [(s_{j-1,k-1} - s_{j-1,k+1})/4 + 1/2]$$

$$s_{j,2k} = [(2s_{j-1,k} - d_{j-1,k} + u_k)/2]$$

$$s_{j,2k+1} = s_{j,2k} + d_{j-1,k} - u_k$$

5/3 变换。

$$d_{j-1,k} = s_{j,2k+1} - [(s_{j,2k+2} - s_{j,2k})/2]$$

$$s_{j-1,k} = s_{j,2k} + [(d_{j-1,k} + d_{j-1,k-1})/4 + 1/2]$$

9/7-M 变换。

$$d_{j-1,k} = s_{j,2k+1} - [(s_{j,2k+4} + s_{j,2k-2}) - 9(s_{j,2k+2} + s_{j,2k})]/16 + 1/2]$$

$$s_{j-1,k} = s_{j,2k} + [(d_{j-1,k} + d_{j-1,k-1})/4 + 1/2]$$

本节主要介绍传统小波（或第一代小波）变换理论的基本知识。首先介绍了连续小波变换和离散小波变换的基本原理；接着讨论了离散小波变换在一维和二维张量积情况下的多分辨分析及相应的 Mallat 算法，Mallat 算法在小波分析中具有非常重要的地位，相当于快速傅里叶变换（FFT）在经典傅里叶分析中的地位；给出了紧支撑双正交小波基的构造方法；最后介绍了提升小波变换（或第二代小波）基本理论。

## 第 22 章 小波变换及其 MATLAB 例程分析

### 22.1 基于小波分析的图像平滑

图像平滑主要是为了消除噪声。噪声并不限于人眼所能看见的失真和变形，有些噪声只有在进行图像处理时才可以发现。图像的常见噪声主要有加性噪声、乘性噪声和量化噪声等。图像中的噪声往往和信号交织在一起，尤其是乘性噪声，如果平滑不当，就会使图像本身的细节如边界轮廓、线条等变得模糊不清。如何既平滑掉噪声又尽量保持图像细节，是图像平滑主要研究的任务。

#### 22.1.1 小波图像平滑的基本原理

一般来说，图像的能量主要集中在其低频部分，噪声所在的频段主要在高频段，因此，如何去掉高频干扰又同时保持边缘信息，是我们研究的内容。

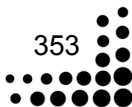
图像平滑的主要目的是为减少噪声。一般情况下，在空间域内可以用平均来减少噪声；而在频率域，由于噪声多在高频段，可以使用各种形式的低通滤波办法来减少噪声。

#### 22.1.2 MATLAB 例程分析

【例 22-1】利用二维小波分析和图像的中值滤波对一给定的含噪图像进行平滑处理。

```
%装载原始图像
load gatin;

%对图像加噪声并显示出含噪图像
init=2788605800;
randn('seed',init);
X=X+10*randn(size(X));
subplot(2,2,1);
image(X);
```





```
colormap(map);
title('含噪图像');

%应用中值滤波进行图像平滑处理
[p,q]=size(X);
for i=2:p-1
    for j=2:q-1
        Xtemp=0;
        for m=1:3
            for n=1:3
                Xtemp=Xtemp+X(i+m-2,j+n-2);
            end
        end
        Xtemp=Xtemp/9;
        X1(i,j)=Xtemp;
    end
end

%显示结果
subplot(2,2,2);
image(X1);
colormap(map);
title('平滑后图像');
```

利用小波变换对图像的平滑处理如图 22-1 所示。

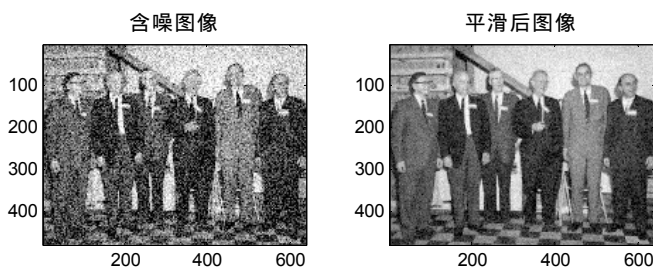


图 22-1 基于小波的图像平滑处理结果

在本例中，我们首先对图像加了一个较大的白噪声，之后应用中值滤波对含噪图像进行处理。从图 22-1 中可以看出，含噪图像经过基于小波的中值滤波处理后具有较好的平滑效果。

## 22.2 基于小波变换数字图像水印研究

数字水印技术的核心就是通过在数字产品中嵌入版权信息以提供产品所有权的证据，任何恶意破坏和去除隐藏信息的手段都将同时导致数字产品被破坏。数字水印信息是嵌在数字



产品中的数字信号,水印的存在要以不破坏原数据的欣赏价值、使用价值为原则。由于水印信息并不影响作品的宏观内容,因而水印信息将永久地保存在多媒体作品当中,任何人若试图从作品中剔除水印都不得不大幅度破坏原作品,从而保护了作者的合法权益。

### 22.2.1 数字水印应具有的特点

目前虽有许多文献讨论数字水印技术,但数字水印始终没有一个明确统一的定义。Cox 等把水印定义为“不可感知地在作品中嵌入信息的操作行为”。不同的应用对数字水印的要求不尽相同,一般认为数字水印应具有如下特点。

(1) 可证明性:水印应能为受到版权保护的信息产品归属提供完全可靠的证据。水印算法能将所有者的有关信息(如注册的用户号码、产品标志或有意义的文字等)嵌入到被保护的對象中,并在需要的时候将这些信息提取。水印可用来判别对象是否受到保护,并能监视被保护数据的传播、真伪鉴别以及非法复制控制等。

(2) 不可感知性:视觉或听觉上的不可感知性,观察者的视觉或听觉系统应察觉不到嵌入水印后载体数据的变化。

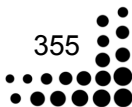
(3) 稳健性:水印应能承受大量有意、无意的物理和几何失真,有意的是指恶意攻击,无意的是指图像压缩、滤波、打印、扫描与复印、噪声污染、尺寸变换等。经过这些操作,稳健的水印算法应仍能从水印载体中提取嵌入的水印或证明水印的存在。一个鲁棒的水印应做到若攻击者试图删除水印将会导致水印载体的彻底破坏。

随着数字水印技术的发展,水印算法的分类方法繁多。按水印的发展,分为第一代和第二代水印;按嵌入的水印信号形式,分为一维和二维水印;按嵌入方法,分为可逆和不可逆水印;按水印检测方法,分为盲水印和非盲水印;按鲁棒性,分为易脆水印、半易脆水印和鲁棒水印;从外观上,分为可见和不可见水印;按载体,分为图像水印、视频水印、音频水印和文档水印;从水印的嵌入域,分为空间域水印和变换域水印。

数字水印有多种用途,可确立版权所有者、认证多媒体来源的真实性、识别购买者、提供关于数字内容的其他附加信息、确认所有权认证和跟踪侵权行为。在篡改鉴定、数据的分级访问、数据跟踪和检测、商业和视频广播、Internet 数字媒体的服务付费、电子商务认证鉴定等方面,它具有十分广阔的应用前景。自 1993 年,该技术引起工业界的浓厚兴趣,并日益成为国际上非常活跃的研究领域,主要应用在以下几个方面。

- (1) 广播监控:通过识别嵌入作品的水印,鉴别作品是何时何地被广播的。
- (2) 所有者鉴别:嵌入代表作品版权所有者身份的水印。
- (3) 所有权验证:发生所有权纠纷时,用水印提供证据。
- (4) 操作跟踪:用水印鉴别合法获取但又非法重新发送内容的人。
- (5) 内容认证:将签名信息嵌入到内容,以待日后检查内容是否被篡改。
- (6) 复制控制:使用水印告知录制设备不能复制的内容。
- (7) 设备控制:使用水印来制造设备。

在 1994 年的 IEEE 国际图像处理会议(ICIP'1994)上,R.G.Schyn del 等人第一次明确提出“数字水印”的概念,掀起了现代信息隐藏技术研究的高潮。ICIP'1996 会议举办了以信息隐藏领域中的水印技术、版权保护和多媒体服务的存取控制为主要内容的研讨专题。同年在





英国剑桥召开第一届信息隐藏国际研讨会，内容涉及数据隐藏、保密通信、密码学等相关学科领域。随着大量学术论文的发表以及在许多重要国际学术会议上设立以信息隐藏技术为主要内容的讨论专题，信息隐藏已成为一门独立的学科领域。在美国，许多著名大学和大公司的研究机构一直在致力于信息隐藏技术方面的研究，并已取得大量研究成果。与此同时，大量数字水印应用软件也应运而生，如图像处理软件 PhotoShop 中的 Digimarc 水印等。

## 22.2.2 数字水印的基本理论框架

一个数字水印方案一般包括 3 个方面：水印的生成、水印的嵌入和水印的提取或检测。数字水印技术实际上是通过水印载体的分析、嵌入信息的预处理、信息嵌入点的选择、嵌入方式的设计、嵌入调制的控制等几个相关技术环节进行合理优化，寻求满足不可感知性、安全可靠、稳健性等诸条件约束下的准最优化设计问题。而作为水印信息的重要组成部分——密钥，则是每个设计方案的重要特色所在，往往可从信息预处理、嵌入点的选择和调制控制等不同环节入手完成密钥的嵌入。

基于变换域的数字水印技术往往采用类似于扩频图像的技术隐藏水印信息。这类技术一般基于常用的图像变换（基于局部或是全局的变换），这些变换包括离散余弦变换（DCT）、离散小波变换（DWT）、傅氏变换（DFT 或 FFT）、傅里叶-梅林（Fourier-Mellin）变换以及哈达马变换（Hadamard transform）等。

实际上，变换域水印算法就是首先利用相应的变换方法（DCT、DWT、DFT 等）将数字图像的空间域数据转化为相应的频域系数；其次，根据待隐藏的信息类型，对其进行适当编码或变形；再次，确定某种规则或算法，用待隐藏的信息的相应数据去修改前面选定的频域系数序列；最后，将数字图像的频域系数经相应的反变换转化为空间域数据。该类算法的隐藏和提取信息操作复杂，隐藏信息量不能很大，但抗攻击能力强，很适合用于数字作品版权保护的数字水印技术。

小波分析是傅里叶分析思想方法的发展与延拓。它既继承和发展了短时傅里叶变换的局部化思想，同时又克服了窗口大小不随频率变化的缺点，是进行信号时频分析、处理时变非稳态信号的比较理想的工具。

小波变换可以在不同尺度下将二维信号（图像）逐层（第  $k$  层）分解为 3 个高频细节子带（水平细节  $HL_k$ 、垂直细节  $LH_k$ 、对角细节  $HH_k$ ）和 1 个低频逼近子带（ $LL_k$ ），逼近子带  $LL_k$  可以按这种方式继续分解成  $HL_{k+1}$ 、 $LH_{k+1}$ 、 $HH_{k+1}$ 、 $LL_{k+1}$ 。低频带表示小波分解在最大尺度、最小分辨率下对原始图像的最佳逼近，它的统计特征和原图像相似，图像的大部分能量集中在此。高频带系列则是图像在不同尺度、不同分辨率下的细节信息。分辨率越低，其中有用信息的比例就越高。也就是说，经过小波分解把一个图像分成了若干级。对于同一级图像，低频子图像  $LL_k$  最重要，其次是  $HL_k$  与  $LH_k$ ，高频子图像  $HH_k$  最不重要。对于不同级来说，级高者重要，级低者不重要。所以，小波图像子频带按其重要总体的排序为  $LL_k$ 、 $HL_k$ 、 $LH_k$ 、 $HH_k$ 、 $HL_{k-1}$ 、 $LH_{k-1}$ 、 $HH_{k-1}$ 、...、 $HL_1$ 、 $LH_1$ 、 $HH_1$ 。图像的离散小波变换 DWT（Discrete Wavelet Transform）后对应各子频带的值为相应的小波系数。根据小波图像各子频带的重要性，针对二值图像将水印信息嵌入到低频逼近系数。

水印的提取过程是嵌入过程的逆过程。对待提取水印的图像进行三色分离，分别得到其



R、G、B 三个色度通道中的图像，然后对 R、G、B 三个色度通道中的图像分别进行 2 级离散小波变换，得到相应的逼近系数和细节系数。同时对原始图像进行三色分离，分别得到其 R、G、B 三个色度通道中的图像，也对其 R、G、B 三个色度通道中的图像分别进行 2 级离散小波变换，得到相应的逼近系数和细节系数。将待提取水印图像和原水印图像对应的 R、G、B 色度通道图像的相应系数分别用水印提取算法提取出水印，再分别进行离散小波反变换，分别得到从 R 通道中提取出的水印图像 1、从 G 通道中提取出的水印图像 2、从 B 通道中提取出的水印图像 3，然后将提取的 3 幅水印图像进行图像融合，得到最佳效果的水印图像。

### 22.2.3 数字水印技术需要解决的问题

数字水印技术是近几年国际学术界兴起的一个前沿研究领域。尽管各种水印算法如雨后春笋般不断涌现，但数字水印技术仍然是一个未成熟的研究领域，还有许多问题需要解决，其理论基础依然非常薄弱，大多数水印算法还是经验性的。主要有以下几个方面还需努力。

(1) 设计对水印系统进行公正的比较和评价方法，在这方面已有部分学者进行一些初步的研究，但缺乏普遍性和原理性，对水印系统的脆弱之处无法进行全面测试与衡量。

(2) 从现实的角度看，水印系统必然要在算法的鲁棒性、水印的嵌入信息量以及不可觉察性之间达到一个平衡，这涉及鲁棒性算法的原理性设计、水印的构造模型、水印能量和容量的理论估计、水印嵌入算法和检测算法的理论研究等方面。如何确定平衡点仍是一个难题，目前大多数水印算法均利用经验而不是从理论上解决此问题。

(3) 如何将水印技术与现行国际图像及视频压缩标准（如 JPEG 2000 和 MPEG-4）相结合，以及如何将水印技术应用于 DVD 工业标准中。

(4) 所有权的证明问题还没有完全解决。就目前已出现的很多算法而言，攻击者完全可破坏掉图像中的水印，或复制出一个理论上存在的“原始图像”，这导致文件所有者不能令人信服地提供版权归属的有效证据。因此，一个好的水印算法应能提供完全没有争议的版权证明，在这方面还需要做很多工作。目前，将水印作为版权保护的 legal 证据还不可能。

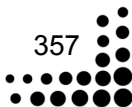
(5) 音频和视频水印的解决方案还不完善，大多数的视频水印算法实际上是将其图像水印的结果直接应用于视频领域中，而没有考虑视频应用中大数据量以及近乎实时的特性。从今后的发展上看，水印在包括 DVD 等数字产品在内的视频和音频领域将有极为广阔的应用前景。因此，如何设计成熟的、合乎国际规范的水印算法仍然悬而未决。

(6) 现有水印算法在原理上有许多雷同之处，但目前国内外的的工作尚未对这些有内在联系的不同算法的共性问题进行高度提炼和深入的理论研究，因而缺乏对数字水印做进一步研究具有指导意义的理论结果。

### 22.2.4 一种基于小波变换的数字水印方法

下面介绍一种典型的基于小波变换的数字水印方法。

(1) 第一步，将水印图像做时域上的变换，目的是对水印信息进行乱序，达到加密的效果。采用函数





$$A_N(k): \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ k & k+1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \mod N$$

其中,  $k$  是一个控制参数;  $N$  是矩阵的大小;  $(x, y)$  和  $(x', y')$  表示像素点在变换前后的位置。假设  $P$  表示由二值水印信息组成的一个  $m \times m$  的矩阵, 对每个点的坐标都做  $A_N(k)$  变换之后, 这个  $m \times m$  的矩阵将变成一个  $N \times N$  的矩阵, 矩阵的每个元素为 0 或 1。

(2) 第二步, 对图像做小波变换, 对变换后得到的小波系数, 选出一个起始位置在  $(P_1, P_2)$ 、大小为  $N \times N$  的系数矩阵。这个矩阵的大小与水印图像作时域变换后形成的矩阵的大小是一致的。

(3) 第三步, 在选出的系数矩阵中嵌入水印信息, 即将两个  $N \times N$  的矩阵进行信息叠加, 其中含有水印信息的矩阵元素为 0 或 1。TYC 提出了一种信息叠加的方案。

设:

$A$  为水印信息进行时域变换后得到的大小为  $N \times N$  的矩阵;

$U$  为在矩阵  $A$  中含有水印信息的位置的集合;

$B$  为图像经过小波变换后得到的系数矩阵 ( $N \times N$ );

$S$  为模;

$C$  为  $B$  和  $U$  的交集;

$\delta(i, j) = c(i, j) \mod S$ 。

对于所有属于  $U$  和  $A$  交集的点  $(i, j)$ , 有

如果  $A(i, j) = 1$ , 并且  $B(i, j) \geq 0$ , 则  $c(i, j) = c(i, j) - \delta(i, j) + T_1$ ;

如果  $A(i, j) = 0$ , 并且  $B(i, j) \geq 0$ , 则  $c(i, j) = c(i, j) - \delta(i, j) + T_2$ ;

如果  $A(i, j) = 1$ , 并且  $B(i, j) < 0$ , 则  $c(i, j) = c(i, j) + \delta(i, j) - T_1$ ;

如果  $A(i, j) = 0$ , 并且  $B(i, j) < 0$ , 则  $c(i, j) = c(i, j) + \delta(i, j) - T_2$ 。

这里,  $T_1$ 、 $T_2$  是水印嵌入的门限; 安全性系数包括  $n$ 、 $k$ 、 $p_1$ 、 $p_2$ 、 $m$ 、 $N$ 、 $S$ 、 $T_1$ 、 $T_2$ 。

水印的提取过程如下。

假设  $Y$  是从小波变换域抽取的一个  $N \times N$  的系数矩阵, 起始位置为  $(P_1, P_2)$ ;  $\theta(i, j)$  满足  $\theta(i, j) = Y(i, j) \mod S$ ;  $D$  是一个  $N \times N$  的矩阵。对  $Y$  中的所有点  $(i, j)$ , 定义:

如果  $|\theta(i, j)| \geq (T_1 + T_2)/2$ , 则  $D(i, j) = 1$ ;

如果  $|\theta(i, j)| < (T_1 + T_2)/2$ , 则  $D(i, j) = 0$ 。

因此, 对矩阵  $D$  做  $T - n$  次  $A_N(k)$  反变换, 水印图像就被恢复出来了。

## 22.2.5 MATLAB 例程分析

【例 22-2】利用二维离散小波变换实现数字水印。

```
clear;
%装入原图像 1
load woman;
I=X;
```



```
%小波函数
type = 'db1';

% 2 维离散 Daubechies 小波变换
[CA1, CH1, CV1, CD1] = dwt2(I,type);
C1 = [CH1 CV1 CD1];

%系数矩阵大小
[length1, width1] = size(CA1);
[M1, N1] = size(C1);

% 定义阈值 T1
T1 = 50;
alpha = 0.2;

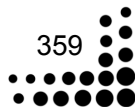
%在图像中加入水印
for counter2 = 1: 1: N1
    for counter1 = 1: 1: M1
        if( C1(counter1, counter2) > T1 )
            marked1(counter1,counter2) = randn(1,1);
            NEWC1(counter1, counter2) = double( C1(counter1, counter2) ) + alpha * abs
                ( double( C1 (counter1, counter2) ) ) * marked1(counter1,counter2) ;
        else
            marked1(counter1, counter2) = 0;
            NEWC1(counter1, counter2) = double( C1(counter1, counter2) );
        end;
    end;
end;

%重构图像
NEWCH1 = NEWC1(1:length1, 1:width1);
NEWCV1 = NEWC1(1:length1, width1+1:2*width1);
NEWCD1 = NEWC1(1:length1, 2*width1+1:3*width1);

R1 = double( idwt2(CA1, NEWCH1, NEWCV1, NEWCD1, type) );

%分离水印
watermark1 = double(R1) - double(I);

figure(1);
subplot(1,2,1);
image(I);
```







```
axis('square');
title('原始图像');
subplot(1,2,2);
imshow(R1/250);
axis('square');
title('Daubechies 小波变换后图像');

figure(2);
imshow(watermark1*10^16);
axis('square');
title('水印图像');

% 水印检测
newmarked1 = reshape(marked1, M1*N1, 1);
% 检测阈值
T2 = 60;
for counter2 = 1: 1: N1
    for counter1 = 1: 1: M1
        if( NEWC1(counter1, counter2) > T2 )
            NEWC1X(counter1, counter2) = NEWC1(counter1, counter2);
        else
            NEWC1X(counter1, counter2) = 0;
        end;
    end;
end;

NEWC1X = reshape(NEWC1X, M1*N1, 1);

correlation1 = zeros(1000,1);
for corrcounter = 1: 1: 1000
    if( corrcounter == 500)
        correlation1(corrcounter,1) = NEWC1X'*newmarked1 / (M1*N1);
    else
        rnmark = randn(M1*N1,1);
        correlation1(corrcounter,1) = NEWC1X'*rnmark / (M1*N1);
    end;
end;

% 计算阈值
originalthreshold = 0;
for counter2 = 1: 1: N1
    for counter1 = 1: 1: M1
        if( NEWC1(counter1, counter2) > T2 )
```



```

        originalthreshold = originalthreshold + abs( NEWC1(counter1, counter2) );
    end;
end;
end;
originalthreshold = originalthreshold * alpha / (2*M1*N1);

corrcounter = 1000;
originalthresholdvector = ones(corrcounter,1) * originalthreshold;

figure(3);
plot(correlation1, '-');
hold on;
plot(originalthresholdvector, '--');
title('原始的加水印图像');
xlabel('水印');
ylabel('检测响应');

```

原始图像和小波变换后的图像如图 22-2 所示,加入的水印图像如图 22-3 所示,水印图像检测结果如图 22-4 所示。

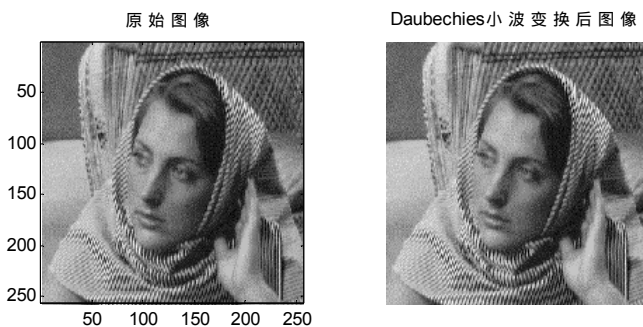


图 22-2 原始图像和小波变换后的图像



图 22-3 水印图像

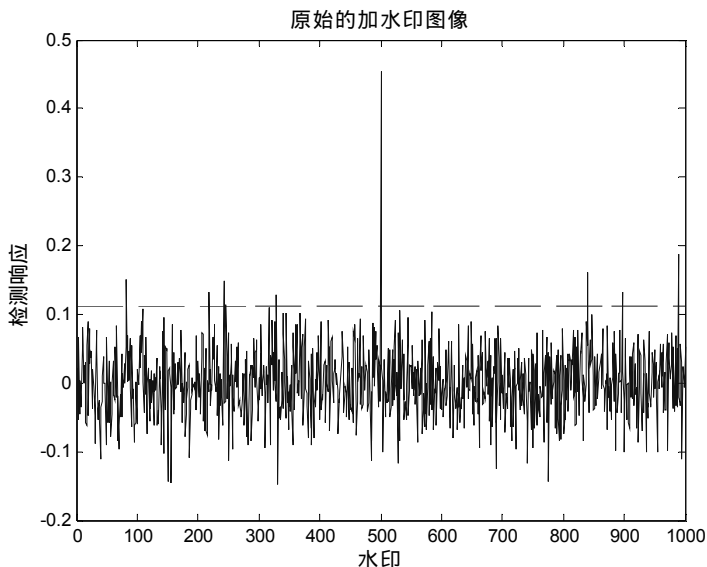


图 22-4 水印图像的检测结果

## 22.3 小波分析与图像增强

数字图像是通过对实际连续的图像按照一定的间隔采样得到的离散化的数据。一般来说，摄影、扫描等成像设备得到的图像质量都不是很高，其中一个重要的原因是可用灰度级没有得到有效使用，图像对比度不够，视觉上比较模糊。为了便于人眼和机器的识别，要求提高图像的对比度，图像增强就是针对这一问题的处理技术。

需要说明的是，增强技术不能增加图像本身所含的信息，但是可以凸显特定的特征，处理更有利于识别，从而有利于特定的应用。通常意义上的图像增强目标主要是放大图像中感兴趣结构的对比度，增加可理解性；同时减少或抑制图像中混有的噪声，提高视觉质量。

### 22.3.1 小波图像增强的基本方法

按照处理空间的不同，常用的增强技术可以分为基于图像域和机遇变幻域两种。前一种方法直接对像素点进行运算，后一种方法相对比较复杂，它首先将图像从空间域变换到另一个域内表示（最常用是时域到频域的变换），通过修正相应域内的系数达到提高输出图像对比度的目的。

增强是图像处理中最基本的技术之一，这里只介绍基于多尺度方法的增强技术。小波变换将一幅图像分解为大小、位置和方向均不相同的分量，在做逆变换之前，可根据需要对不同位置、不同方向上的某些分量改变其系数的大小，从而使得某些感兴趣的分量放大而使某些不需要的分量减小。

小波分析在二维信号（图像）处理方面的优点主要体现在其时频分析特性。前面介绍了一些基于这种特性的应用的实例，但对二维信号小波系数的处理方法只介绍了阈值化方法一



种,下面将介绍以前在一维信号中用到的抑制系数的方法,这种方法在图像处理领域主要应用于图像增强。

图像增强问题的基本目标是对图像进行一定的处理,使其结果比原图像更适用于特定的应用领域。这里,“特定”这个词非常重要,因为几乎所有的图像增强问题都是与问题背景密切相关的,脱离了问题本身的知识,图像处理的结果可能并不一定适用。比如,某种方法可能非常适用于处理 X 射线图像,但同样的方法可能不一定也适用于火星探测图像。

在图像处理领域,图像增强问题主要通过时域处理(沿用信号处理的说法,空域可能对图像更适合)和频域处理两种方法来解决。时域方法通过直接在图像点上作用算子或掩码来解决,频域方法通过修改傅里叶变换系数来解决。这两种方法的优劣很明显,时域方法方便快捷但会丢失很多点之间的相关信息,频域方法可以很详细地分离出点之间的相关,但需要做两次数量级为  $n\log_2 n$  的傅里叶变换和逆变换的操作,计算量大得多。

小波分析是以上两种方法的权衡结果,建立在如下的认识基础上;傅里叶分析在所有点的分辨率都是原始图像的尺度,但对于问题本身的要求,我们可能不需要这么高的分辨率,而单纯的时域分析又显得太粗糙。小波分析的多尺度分析特性为用户提供了更灵活的处理方法,可以选择任意的分解层数,用尽可能少的计算量得到我们满意的结果。

小波变换将一幅图像分解为大小、位置和方向都不同的分量。在做逆变换之前可以改变小波变换域中某些系数的大小,这样就能够有选择地放大所感兴趣的分量而减小不需要的分量。

## 22.3.2 图像增强的 MATLAB 例程

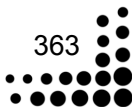
### 1. 图像增强

给定一个 wmandril.mat 图像信号。由于图像经二维小波分解后,轮廓主要体现在低频部分,细节部分体现在高频部分,因此可以通过对低频分解系数进行增强处理,对高频分解系数进行衰减处理,从而达到图像增强的效果。

**【例 22-3】**图像增强应用实例。

具体程序清单如下:

```
Load flujet;  
%画出原始图像  
subplot(221);image(X);colormap(map);  
title('原始图像','fontsize',8);  
axis square  
%下面进行图像的增强处理  
%用小波函数 sym4 对 X 进行 2 层小波分解  
[c,s]=wavedec2(X,2,'sym4');  
sizec=size(c);  
%对分解系数进行处理以突出轮廓部分,弱化细节部分  
for i=1:sizec(2)  
    if(c(i)>350)  
        c(i)=2*c(i);
```





```
else
    c(i)=0.5*c(i);
end
end
%下面对处理后的系数进行重构
xx=waverec2(c,s,'sym4');
%画出重构后的图像
subplot(222);image(xx);
colormap(map);
title('增强图像','fontsize',8);
axis square
```

输出结果如图 22-5 所示。

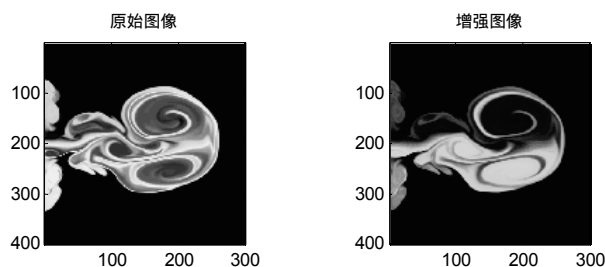


图 22-5 小波分析用于图像增强

下面将主要讨论图像增强中的钝化和锐化两种方法。钝化操作主要是提出图像中的低频成分，抑制尖锐的快速变化成分；锐化操作正好相反，将图像中尖锐的部分尽可能提取出来，用于检测和识别等领域。下面将举例说明这两种方法在 MATLAB 中的实现，并对于基于傅里叶变换的传统频域方法同小波方法做比较。

## 2. 图像钝化

图像钝化在时域中的处理相对简单，只需要对图像作用一个平滑滤波器，使得图像中的每个点都与其相邻点做平滑即可，这里不做详细介绍。我们来介绍基于傅里叶变换的频域处理方法。

下面以 chess 信号为例，通过两种方法对图像钝化的结果做一下比较。

### 【例 22-4】图像钝化实例。

```
% 读入 chess 信号
load chess
% 分别保存用 DCT 方法和小波方法的变换系数
blur1=X;
blur2=X;
% 对原图像做二维离散余弦变换
ff1=dct2(X);
% 对变换结果在频域做 BUTTERWORTH 滤波
```



```

for i=1:256
    for j=1:256
        ff1(i,j)=ff1(i,j)/(1+((i*j+j*j)/8192)^2);
    end
end
% 重建变换后的图像
blur1=idct2(ff1);
% 对图像做 2 层的二维小波分解
[c,l]=wavedec2(X,2,'db3');
csize=size(c);
% 对低频系数进行放大处理，并抑制高频系数
for i=1:csz(2);
    if(c(i)>300)
        c(i)=c(i)*2;
    else
        c(i)=c(i)/2;
    end
end
% 通过处理后的小波系数重建图像
blur2=waverec2(c,l,'db3');
% 显示三幅图像
subplot(221);
image(wcodemat(X,192));
colormap(gray(256));
title('原始图像','fontsize',8);
subplot(223);
image(wcodemat(blur1,192));
colormap(gray(256));
title('采用 DCT 方法钝化图像','fontsize',8);
subplot(224);
image(wcodemat(blur2,192));
colormap(gray(256));
title('采用小波方法钝化图像','fontsize',8);

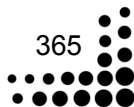
```

在命令行中运行该程序后，得到如图 22-6 所示的显示结果。

从图 22-6 中可以看出，采用 DCT 在频域做滤波的方法得到钝化结果更为平滑，这是因为其分辨率最高；而小波方法得到的结果在很多地方有不连续的现象，因为我们对系数做放大或抑制在阈值两侧有间断，而且分解层数很少，没有完全分离出频域的信息。另外，我们在做系数放大或抑制的时候，采用的标准是根据系数绝对值的大小，没有完全体现出其位置信息，但是在小波系数中，我们很容易在处理系数的过程中加入位置信息。

### 3. 图像锐化

与图像钝化所做的工作相反，图像锐化的任务是突出高频信息、抑制低频信息，从快速变





化的成分中分离出标识系统特性或区分系统边界的成分,以便于进一步的识别、分割等操作。

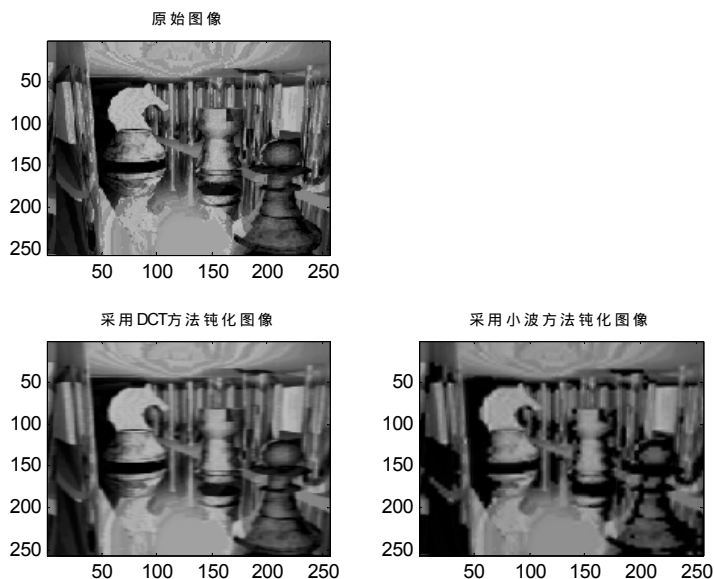


图 22-6 图像钝化

在时域(空域)中,锐化的方法不外乎是作用掩码或做差分。同钝化的道理一样,无论是掩码和差分都很难识别点之间的关联信息。下面的例子同样是在频域完成的,用传统的傅里叶分析方法(这里采用的是 DCT 变换)得到的频域系数。

【例 22-5】图像锐化实例。

```
% 读入 chess 信号
load chess;
% 分别保存用 DCT 方法和小波方法的变换系数
blur1=X;
blur2=X;
% 对原图像做二维离散余弦变换
ff1=dct2(X);
% 对变换结果在频域做 BUTTERWORTH 滤波
for i=1:256
    for j=1:256
        ff1(i,j)=ff1(i,j)/(1+(32768/(i*i+j*j))^2);
    end
end
% 重建变换后的图像
blur1=idct2(ff1);
% 对图像做 2 层的二维小波分解
[c,l]=wavedec2(X,2,'db3');
csize=size(c);
```

% 对低频系数进行放大处理，并抑制高频系数

```
for i=1:size(2);
    if(abs(c(i))<300)
        c(i)=c(i)*2;
    else
        c(i)=c(i)/2;
    end
end
% 通过处理后的小波系数重建图像
blur2=waverec2(c,l,'db3');
% 显示 3 幅图像
subplot(221);
image(wcodemat(X,192));
colormap(gray(256));
title('原始图像','fontsize',8);
subplot(223);
image(wcodemat(blur1,192));
colormap(gray(256));
title('采用 DCT 方法锐化图像','fontsize',8);
subplot(224);
image(wcodemat(blur2,192));
colormap(gray(256));
title('采用小波方法锐化图像','fontsize',8);
```

得到的结果如图 22-7 所示。

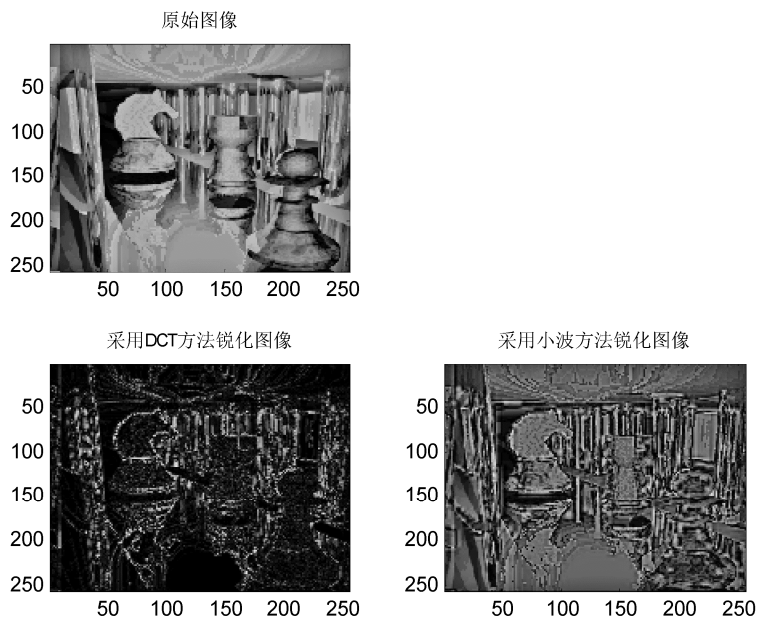


图 22-7 图像锐化





从图 22-7 中可以看出,使用 DCT 方法进行高通滤波得到的高频结果比较纯粹,完全是原图像上的边缘信息;而在小波方法得到的结果中,不只有高频成分,还有变换非常缓慢的低频成分,这是因为两者同样在小波系数上体现为绝对值较低的部分,但这些成分的存在对我们进行进一步分析并无多大影响。

最后,我们来比较一次这两个例子的时间复杂度,对 DCT 方法,需要做两次复杂度为  $O(n \log_2 n)$  的 DCT 变换,中间系数处理部分复杂度为  $O(n)$ ,而对小波变换,无论是分解和重构,系数处理的复杂度都是  $O(n)$ ,所以时间复杂度的优势非常明显。

## 22.4 小波分析与图像融合

基于小波分析的融合方法是 Mallat 于 1989 年提出小波的多分辨率分析思想及小波的分解和重构快速算法。目前,小波分析的研究比较热门,在遥感影像融合处理中也经常用到小波分析理论。小波融合方法的基本函数是一些小型波,所以称为小波融合。在影像数据融合中,小波变换可将影像分解为更低分辨率的近似低频影像和高频细节影像。同时由于小波变换的多分辨率特性,不同尺度的空间特征也可以进行分离。因此,小波变换可以用于不同传感器间多分辨率影像的融合。

### 22.4.1 小波图像融合的基本原理

目前基于小波变换的图像数据融合方法特别多。霍宏涛等人曾针对光谱扭曲这一问题,提出了一种直接相加的小波变换低频信息处理方法,即将未分解的 TM 放大图像与小波分解后的 SPOT 图像的高频部分重构,得到的图像通过直接相加来生成融合图像。用该方法得到的融合图像,它的光谱扭曲值和信息量都优于传统的小波融合方法,但在解决彩色畸变方面还不令人满意。为了提高融合图像的质量,蒋晓瑜、王文杰等人在基于传统小波变换融合的实验基础上分析认为,简单地用低分辨率图像的低频部分来代替高分辨率图像的低频部分将会造成原低分辨率图像的信息损失,特别是在两种传感器图像相关性较差时,会对融合结果有较大的影响,传统的小波融合在融合时存在光谱扭曲,为了减少在融合时产生的光谱扭曲,从而提出了基于区域能量特征的融合算法。李军、强赞霞等人则认为方差是一个反差的测度,它的大小体现了图像细节的信息量,因此可将方差作为融合的依据,提出了采用局域方差准则的小波变换融合法,他们提出的优化融合方法在解决传统小波融合方法的光谱扭曲方面起到了一定的作用。而吴兆福等人在他们的基础上又提出了基于边缘特征的遥感图像小波变换融合法,采用 ETM+图像与 SPOT 图像。研究表明,这种方法优于前面提到的局域方差准则与区域能量特征的融合方法。

图像融合是将同一对象的两个或更多的图像合成在一幅图像中,以便它比原来的任一幅图像更容易得为人们所理解。这一技术可应用于多频谱图像理解以及医学图像处理等领域。在这些场合,同一物体部件的图像往往是采用不同的成像机理得到的。

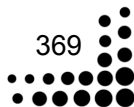


## 22.4.2 MATLAB 例程分析

### 【例 22-6】图像融合例程。

下面用二维小波分析将例 22-5 中的 woman.mat 和 wbarb.mat 两幅图像融合在一起。

```
load woman;
X1=X;map1=map;
%画出原始图像
subplot(131);image(X1);colormap(map1);
title('woman');
axis square
load wbarb;
X2=X;map2=map;
for i=1:256
    for j=1:256
        if (X2(i,j)>100)
            X2(i,j)=1.2*X2(i,j);
        else
            X2(i,j)=0.5*X2(i,j);
        end
    end
end
subplot(132);
image(X2);
colormap(map2);
title('wbarb');
axis square
%用小波函数 sym4 对 X1 进行 2 层小波分解
[c1,s1]=wavedec2(X1,2,'sym4');
%对分解系数进行处理以突出轮廓部分，弱化细节部分
sizec1=size(c1);
for i=1:sizec1(2)
    c1(i)=1.2*c1(i);
end
%用小波函数 sym4 对 X2 进行 2 层小波分解
[c2,s2]=wavedec2(X2,2,'sym4');
%下面进行小波变换域的图像融合
c=c1+c2;
%减小图像亮度
c=0.5*c;
%对融合的系数进行重构
xx=waverec2(c,s1,'sym4');
%画出融合后的图像
```





```
subplot(133);image(xx);
title('融合图像','fontsize',8);
axis square
```

程序运行结果如图 22-8 所示。

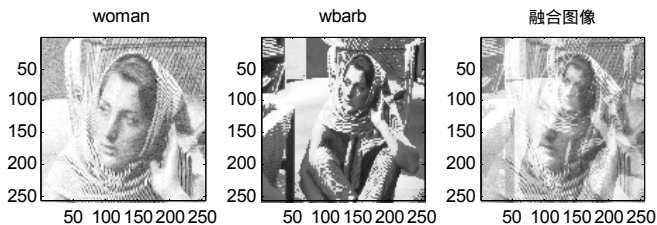


图 22-8 小波分析用于图像融合

图 22-8 中，一幅图像和它某一部分放大后的图像融合，融合后的图像给人一种朦朦胧胧梦幻般的感觉，对较深的背景部分则做了淡化处理。

**【例 22-7】**利用图像融合方法从模糊图像中恢复图像。

```
%调入第一幅模糊图像
load chess;
X1=X;
%调入第二幅模糊图像
load chess;
X2=X;
%基于小波分解的图像融合
XFUS=wfusing(X1,X2,'sym4',5,'max','max');
%显示
colormap(map);
subplot(2,2,1);
image(X1);
axis square;
title('模糊图 1');
subplot(2,2,2);
image(X2);
axis square;
title('模糊图 2');
subplot(2,2,3);
image(XFUS);
axis square;
title('恢复后图像');
```

利用图像融合从两幅模糊的原始图像中恢复出原图像的结果如图 22-9 所示。

小波分析的应用是与小波分析的理论研究紧密地结合在一起的。现在，它已经在科技信息领域取得了令人瞩目的成就。目前，对性质随时间稳定不变的信号，处理的理想工具仍然是傅里叶分析。但在实际应用中，绝大多数信号是非稳定的，小波分析正是适用于非稳定信



号的处理工具。图像处理是针对性很强的技术，根据不同应用、不同要求需要采用不同的处理方法。采用的方法是综合各学科较先进的成果而成的，如数学、物理学、心理学、信号分析学、计算机学、和系统工程等。计算机图像处理主要采用两大类方法：一类是空域中的处理，即在图像空间中对图像进行各种处理；另一类是把空间与图像经过变换，如傅里叶变换，变换到频率域，在频率域中进行各种处理，然后再变换回图像的空间域，形成处理后的图像。图像处理是“信息处理”的一个方面，这一观点现在已经为人所熟知。它可以进一步细分为多个研究方向；如图片处理、图像处理、模式识别、景物分析、图像理解、光学处理等。小波分析在图像处理方面，主要是用来进行图像压缩、图像去噪、图像增强（包括图像钝化和图像锐化）、图像融合、图像分解。

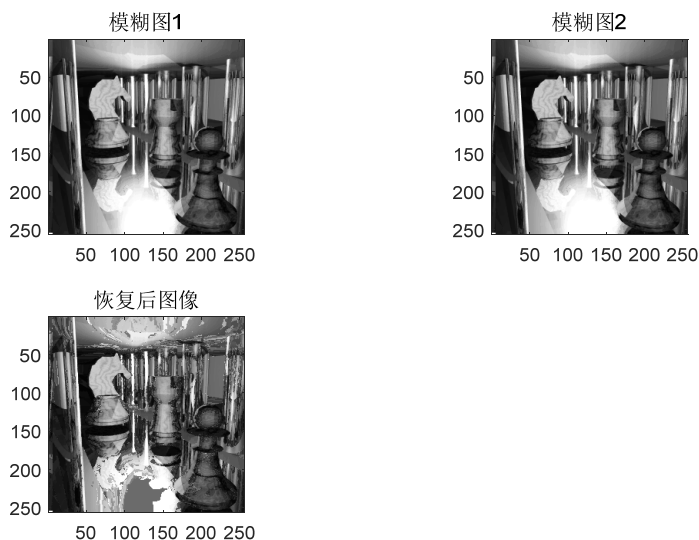
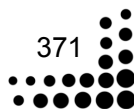


图 22-9 利用图像融合恢复图像



# 附录 A MATLAB 的程序设计及绘图功能



## A.1 MATLAB 程序设计原则

MATLAB 程序的基本设计原则如下所述。

- (1) 百分号“%”后面的内容为程序的注释，要善于运用注释使程序更具可读性。
- (2) 养成在主程序开头用 clear 指令清除变量的习惯，以消除工作空间中其他变量对程序运行的影响，但注意在子程序中不要用 clear。
- (3) 参数值要集中放在程序的开头部分，以便维护。要充分利用 MATLAB 工具箱提供的指令来执行所要进行的运算，在语句行之后输入分号使其及中间结果不显示在屏幕上，以提高运行速度。
- (4) input 函数可以用来输入一些临时的数据；而对于大量参数，则通过建立一个存储参数的子程序，在主程序中通过子程序的名称来调用。
- (5) 程序尽量模块化，即采用主程序调用子程序的方法，将所有子程序合并在一起来执行全部的操作。
- (6) 充分利用 Debugger 来进行程序的调试（设置断点、单步执行、连续执行），并利用其他工具箱或图形用户界面（GUI）的设计技巧，将设计结果集成到一起。
- (7) 设置好 MATLAB 的工作路径，以便程序的调用。

## A.2 M 文件

在 MATLAB 中，以文件形式保存的源代码一般使用“.m”作扩展名，这样的文件称为 M 文件。M 文件有两种，一种是脚本文件，另一种是函数文件。编辑 M 文件时，需要打开编辑、调试窗口，有以下几种打开方法。

- 单击 MATLAB 工具栏上的“新建脚本”图标。
- 直接在命令行窗口输入“edit”命令。

M 文件编辑窗口如图 A-1 所示。

可以看到，编辑窗口有自己的菜单和工具栏，不仅可以实现对文件的管理功能，实际上还可以直接进行程序调试。

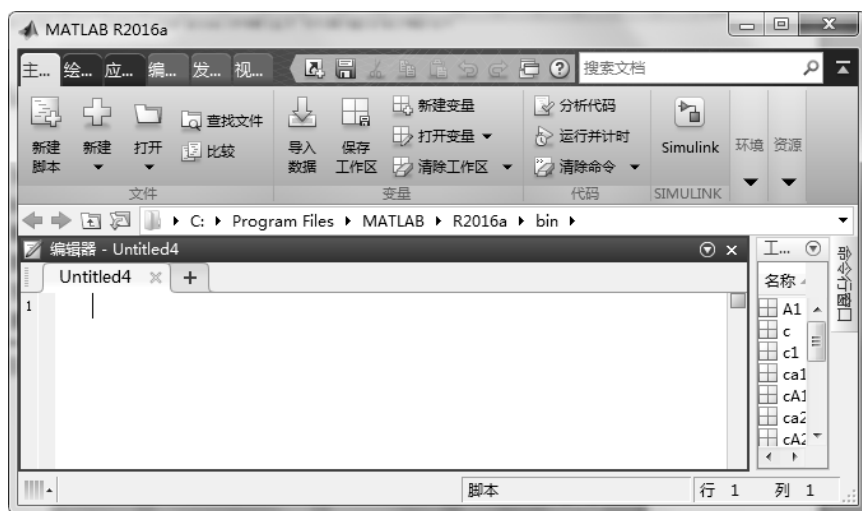


图 A-1 M 文件编辑窗口

### 1. 脚本 M 文件

脚本 M 文件是许多 MATLAB 代码按顺序组成的命令序列集合，不接受参数的输入与输出，与 MATLAB 工作空间共享变量空间。

脚本 M 文件一般用来实现一个相对独立的功能，如对某个数据集进行某种分析、绘图，求解某个已知条件下的微分方程等。用户可以通过在命令窗口中直接输入文件名来运行脚本 M 文件。

通过脚本 M 文件，用户可以把实现一个具体功能的一系列 MATLAB 代码书写在一个 M 文件中，并保存它，每次只需要输入该文件名即可运行脚本 M 文件中的所有代码。

【例 A-1】建立脚本 M 文件，使其功能为求 100！

```
%以下代码用以实现 100！
s=1;
for i=2:100      %开始 for 循环
    s=s*i;
end
disp('100 的阶乘为：');
s
```

把以上文件保存为 li2\_1.m。在命令窗口中输入“li2\_1”，并按 Enter 键，结果如下：

```
100 的阶乘为：
s =
    9.3326e+157
```

用 whos 来查看运行后内存中的变量，例如：

```
>> whos

Name      Size      Bytes      Class      Attributes
i          1 × 1         8         double
s          1 × 1         8         double
```



```
>> clear      %清除内存中的变量，之后再运行 whos 将什么也不显示
>> whos      %不显示任何内容
```

M 文件保存时要注意以下几点。

(1) 文件名一律以字母开头，以字母、数字或下画线组成，不要含有空格、减号等。例如，1.m、egB-1.m、eg2.1.m 等都是不合法的。

(2) M 文件一般都用小写字母，尽管 MATLAB 区分变量的大小写，但并不区分文件名的大小写。例如，A 与 a 是不同的变量，但 EG2\_1.m 与 eg2\_1.m 是相同的。

(3) 要防止它与变量名冲突。为此，变量名一般用一两个字符（如 a、a1）表示，而 M 文件名一般用 4 个以上字符，如 eg2\_1.m、prog1 等。另外，系统内部的保留字及内部 M 文件名也不要，如 function、while、clear 等。

(4) MATLAB 只执行已保存的 M 文件，所以不要忘记每次修改程序后都要保存。

(5) M 文件一般保存在当前目录，否则很可能得不到执行结果。

## 2. 函数 M 文件

函数 M 文件也是为了实现一个单独功能的代码块，但与脚本 M 文件不同的是函数 M 文件需要接受参数的输入与输出，函数 M 文件中的代码一般只处理输入参数传递的数据，并把处理结果作为函数输出参数返回给 MATLAB 工作空间中的指定接收变量。

MATLAB 提供的许多函数就是用函数 M 文件编写的，尤其是各种工具箱中的函数，用户可以打开这些 M 文件来查看。实际上，对于特殊应用领域的用户，如果积累了充分的专业领域应用函数，就可以组建自己的专业领域工具箱了。

函数 M 文件与脚本 M 文件的主要区别有以下 3 点。

(1) 由 function 开始，后跟的函数名必须与文件名相同。

(2) 有输入/输出变元（变量），可进行变量传递。

(3) 除非用 global 声明，程序中的变量均为局部变量，不保存在工作空间中。

一个完整的 M 函数文件包括如下 5 部分结构：

```
function t = date                                %函数声明行
%DATE      Current date as date string.          %H1 行
%   S = DATE returns a string containing the date in dd-mmm-yyyy format.
%
%   See also NOW, CLOCK, DATENUM.

%   Copyright 1984-2002 The MathWorks, Inc.
%   $Revision: 1.15 $   $Date: 2002/04/15 03:20:04 $   %帮助文字

c = clock;                                       %在此行开始到最后一行都为正文内容
mths = ['Jan','Feb','Mar','Apr','May','Jun','Jul';
        'Aug','Sep','Oct','Nov','Dec'];
d = sprintf('%0f',c(3)+100);
t = [d(2:3) '-' mths(c(2),:) '-' sprintf('%0f',c(1))];
```

• 函数声明行（Function Definition Line）：这行只出现在函数 M 文件的第一行，通过



function 关键字表明此文件是一个函数 M 文件，并指定函数名、输入与输出参数。

- H1 行：这是帮助文字的第一行 (the First Help Text Line)，给出 M 文件帮助信息最关键的信息。当用 lookfor 查找某个函数相关的函数时，lookfor 只在 H1 行中搜索是否出现指定函数。
- 帮助文字：这部分对 M 文件有更加详细的说明，经常解释 M 文件实现的功能，M 文件中出现的各变量、参数意义，以及创作版权信息等。当获取一个 M 文件的帮助时，H1 行与帮助文字部分同时显示。
- 正文内容：这时 M 文件实现功能的 MATLAB 代码部分，通常包括运算、赋值等指令。
- 注释部分：这部分出现的位置比较灵活，主要是用来注释 M 文件正文的具体运行过程，以方便阅读与修改，经常穿插在 M 文件正文中间。

注意：当函数具有多个输出变量时，以方括号括起；当函数具有多个输入变量时，则直接用圆括号括起，如 `function [x, y, z]=sphere(theta, phi, rho)`。当函数不含输出变量时，则直接略去输出部分或采用空方括号表示，如 `function printresults(x)`；`function []=printresults(x)`。

## A.3 MATLAB 的流程控制

与各种常见的高级语言一样，MATLAB 也提供了多种经典的流程控制语句。MATLAB 中的程序流程控制语句有顺序结构 (input、disp)、分支结构 (if 与 switch 结构)、循环结构 (for、while 循环)、错误控制结构 (try-catch 结构)、其他流程控制 (continue、break、return 语句)，下面分别进行介绍。

### 1. 顺序结构

顺序结构是指按照程序中的语句排列顺序依次执行，直到最后一个语句。这是最简单的一种程序结构。一般涉及数据的输入、数据的计算或处理、数据的输出等内容。

(1) 数据的输入。从键盘输入数据，可以使用 input 函数来进行，该函数的调用格式为：

```
A=input(提示信息, 选项)
```

其中，提示信息为一个字符串，用于提示用户输入什么样的数据。例如，从键盘输入 A 矩阵，可以采用下面的命令来完成：

```
A=input('请输入一个有效数字：')
```

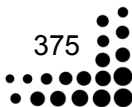
执行该语句时，首先在屏幕上显示提示信息“请输入一个有效数字：”，然后等待用户从键盘输入一个有效数字。

如果在 input 函数调用时采用 's' 选项，则允许用户输入一个字符串。例如，想输入一个人的姓名，可采用如下命令：

```
a=input('请输入邮箱号','s');
```

(2) 数据的输出。MATLAB 提供的命令窗口输出函数主要有 disp、fprintf 函数。disp 函数的调用格式为：

```
disp(输出项)
```







其中，输出项既可以为字符串，也可以为矩阵。例如：

```
>> A='欢迎您!';  
>> disp(A)
```

输出为：

```
欢迎您!  
又如：  
>> B=rand(3,4);      %创建一个随机矩阵  
>> disp(B)  
    0.8147    0.9134    0.2785    0.9649  
    0.9058    0.6324    0.5469    0.1576  
    0.1270    0.0975    0.9575    0.9706
```

以下举例说明 fprintf 函数最常见的使用方式。

如果输入命令：

```
fprintf('圆周率 pi=%10.8f',pi)
```

则会按浮点型输出含 8 位小数、1 位整数的圆周率近似值，其输出结果为：

```
圆周率 pi=3.14159265
```

如果输入命令：

```
>> a=23.28;fprintf('a=%d',a)
```

则会按整型数输出 a 的值，输出为：

```
a=2.328000e+001
```

如果输入命令：

```
>>a=23.28;fprintf('a=%f',a)
```

则会按浮点型数输出 a 的值，输出为：

```
a=23.280000
```

## 2. 分支结构

分支结构可以使程序中的一段代码只在满足一定条件时才执行，因此也成为分支选择。MATLAB 中分支语句有两类，下面分别进行介绍。

(1) if 语句。if 与 else 或 elseif 连用，偏向于是非选择，当某个逻辑条件满足时执行 if 后的语句，否则执行 else 语句。

if 结构的语法格式为：

```
if expression1  
    statements1  
elseif expression2  
    statements2  
else  
    statements3  
end
```



其中,elseif 与 else 语句都是可选语句。if、elseif 与 else 构成的各项分支里,哪个条件满足(表达式 expression 的结果为真),就执行哪个分支后面紧跟的程序语句。因此,各个分支条件满足的情况应该是互斥的与完全的,就是被选的条件在一个分支中成立,而且只能在一个分支中成立。

当然,若省略了 elseif 与 else 分支的语句,就不必要求分支条件满足的情况具备完全性了。

if 结构中的条件判断除了可以用表达式外,还可以用数组 A,这时的判断相当于表达式 all(A),即当数组 A 的所有元素都为非零值时,才执行该条件后的分支代码。

特别地,当数组 A 为空数组时,相当于该条件判断为假。

【例 A-2】利用 if 语句创建一个数组。

```
>> for m = 1:5
    for n = 1:5
        if m == n
            a(m,n) = 2;
        elseif abs(m-n) == 2
            a(m,n) = 1;
        else
            a(m,n) = 0;
        end
    end
end
a
```

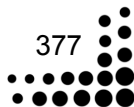
程序运行结果如下:

```
a =
     2     0     1     0     0
     0     2     0     1     0
     1     0     2     0     1
     0     1     0     2     0
     0     0     1     0     2
```

(2) switch 语句。switch 与 case、otherwise 连用,偏向于情况的列举,当表达式结果为某个或某些值时,执行特定 case 指定的语句块,否则执行 otherwise 语句。

switch 结构语法格式为:

```
switch switch_expr
case case_expr
    statement, ..., statement
case {case_expr1, case_expr2, case_expr3, ...}
    statement, ..., statement
otherwise
    statement, ..., statement
end
```





执行中，先计算表达式 `switch_expr` 的值，当结果等于某个 `case` 的 `case_expr` 时，就执行该 `case` 紧随的语句；如果所有 `case` 的 `case_expr` 都与 `switch_expr` 计算结果不相等，则执行 `otherwise` 后面紧随的语句。

`otherwise` 语句是可选的，当没有 `otherwise` 语句时，如果所有 `case` 的 `case_expr` 都与 `switch_expr` 计算结果不相等，则跳过 `switch-case` 语句段，直接执行后续语句。

相等的意义，对于数值类型来说，相当于判断 `if result = case_expr`；对于字符串类型来说，相当于判断语句 `if strcmp(result, case_expr)`。

由此可见，`switch-case` 语句实际上可以被 `if-elseif-else` 语句等效替换，不过两种结构各有更便利的地方，可通过下面两段代码来体会：

```
if (x == 1) {
    y = 1;
} else if (x == 2) {
    y = 2;
} else if (x == 3) {
    y = 3;
} else {
    y = 4;
}
end

switch (x) {
    case 1:
        y = 1;
    case 2:
        y = 2;
    case 3:
        y = 3;
    default:
        y = 4;
}
end
```

**【例 A-3】**检查变量 `x` 的值。如果 `x` 等于 -1、0 或 1，那么以文本的形式打印 `x` 的值；如果 `x` 不等于这 3 个值中的任何一个，则执行 `otherwise` 语句中的 `other value`。代码如下：

```
function example_case(x)
switch x
    case -1
        disp('输入值为-1');
    case 0
        disp('输入值为 0');
    case 1
        disp('输入值为 1');
    otherwise
        disp('输出的为其他值');
end
```

在一条 `case` 语句后可以列举多个值，只需要以元胞数组的形式列举多个值，也就是用花括号把用逗号或空格分隔的多个值括起来。例如：

```
switch var
    case 1
        disp('输入值为 1');
    case {2,3,4}          %判断多个输入值
        disp('输入值为 2, 3, 4 中的一个');
    case 5
```



```

        disp('输入值为 5');
    otherwise
        disp('输入其他的值');
    end

```

### 3. 循环结构

循环控制语句能够使得某段程序代码多次重复执行，MATLAB 中提供了两类循环语句，分别为 for 与 while 循环。下面分别进行介绍。

(1) for 循环。for 循环一般用在已知循环执行次数的情况。其结构语句格式为：

```

for index = start:increment:endvalue
    program statements
    :
end

```

其中，index 为循环变量；increment 为增量；endvalue 用于判断循环是否应终止。增量 increment 默认值为 1，可以自由设置。当增量为正数时，循环开始先将 index 赋值为 start，然后判断 index 是否小于等于 endvalue，若是，则执行循环语句，执行完后，对 index 自增一个增量 increment；再判断 index 是否小于等于 endvalue，若是，则继续执行循环语句，继续对 index 进行自增，循环反复进行，直到 index 大于 endvalue 时退出循环。

增量 increment 也可以设置为负整数，表示每次循环执行后对循环变量 index 进行自减一个增量值 increment，当 index 小于 endvalue 时，退出循环。

for 循环中的循环变量 index 也可以赋值为数组 A，那么在第一次循环中 index 就被赋值为 A(:,1)，即 A 的第一列元素，第二次循环 index 被赋值为 A(:,2)，依次类推；若 A 有 n 列元素，则循环执行 n 次，第 n 次循环时，循环变量 index 被赋值为 A(:,n)。

【例 A-4】利用多重嵌套循环给矩阵各元素赋值。

```

>> for m=1:5
    for n=1:5
        a(m,n)=1/(m+n+1);
    end
end
a

```

程序运行结果如下：

```

a =
    0.3333    0.2500    0.2000    0.1667    0.1429
    0.2500    0.2000    0.1667    0.1429    0.1250
    0.2000    0.1667    0.1429    0.1250    0.1111
    0.1667    0.1429    0.1250    0.1111    0.1000
    0.1429    0.1250    0.1111    0.1000    0.0909

```

(2) while 语句。while 循环用在已知循环退出条件的情况。其结构语法格式为：

```

while expression
    program statements
end

```



```
        :  
    end
```

当表达式 expression 的结果为真时，就执行循环语句，直到表达式 expression 的结果为假时，循环停止。

【例 A-5】利用 while 循环结构求出  $\sum_{i=0}^{68} (3i-1)$  的值。

```
>> n=68;sum=0;  
i=0;  
while i<=n  
    sum=sum+(3*i-1);  
    i=i+1;  
end  
sum
```

程序运行结果如下：

```
sum =  
    6969
```

#### 4. 错误控制结构

错误控制结构使用由 try-catch 语句，用来捕捉并处理异常。try 语句用来检测程序代码是否会产生错误，一旦错误发生，MATLAB 会立即跳入相应的 catch 语句中去，对错误做相应的处理。其结构格式为：

```
try  
    program statements  
    :  
catch exception  
    error-handling statements  
    :  
end
```

在程序不出错的情况下，这种结构只有语句 program statements 被执行；若程序出现错误，那么错误信息将被捕获，并存放在 lasterr 变量中，然后执行 error-handling statements；若在执行语句组时，程序又出现错误，那么程序将自动终止，除非相应的错误信息被另一个 try-catch 结构所捕获。

【例 A-6】用于检验实现 100 的阶乘代码是否出错。

```
>> try  
    i=1;  
    s=1;  
    while i<=100  
        s=s*i;  
        i=i+1;  
    end
```



```
disp('100 的阶乘为：');
S      %S 变量为出错代码
catch
    disp('程序出现错误！')
    disp('');
    disp('错误为：');
    lasterr
end
```

程序运行结果如下：

```
100 的阶乘为：
程序出现错误！
错误为：
ans =
Undefined function or variable 'S'.
```

从例 A-6 可清楚地看到 try-catch 结构的运行顺序，先逐行运行 try 与 catch 之间的语句，当运行到出错处时，系统将这一错误信息捕获并将其保存到变量 lasterr 中，然后执行 catch 与 end 之间的程序代码。

## 5. 其他流程控制

在利用 MATLAB 编程解决实际问题时，可能会需要提前终止 for 与 while 等循环结构，有时可能需要显示必要的出错或警告信息等。下面分别对实现这些特殊要求所需的流程控制命令进行介绍。

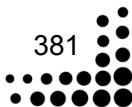
(1) continue 命令。continue 在 for 或 while 循环中用于直接跳到下一个循环的执行。在嵌套式循环中，continue 控制的是与自己最近的一个 for 或 while 循环。

【例 A-7】计算文件 magic.m 的行数，空白行与注释行除外。continue 语句相当于在遇到空白行或注释行时，继续读入 magic.m 中的下一行而不增加行数值。代码如下：

```
fid = fopen('magic.m','r');
count = 0;
while ~ feof(fid)
    line = fgetl(fid);
    if isempty(line) || strncmp(line,'% ',1) || ~ ischar(line)
        continue
    end
    count = count + 1;
end
fprintf('%d lines\n',count);
fclose(fid);
```

显示为文件 magic.m 的行数。

(2) break 命令。break 命令一般用来终止 for 或 while 循环，通常与 if 条件语句一起使用，如果条件满足则利用 break 命令将循环终止。在多层循环嵌套中，break 只终止最内层的循环。





【例 A-8】如果满足循环条件即利用 break 退出循环。

```
>> a = 0; fa = -Inf;
b = 3; fb = Inf;
while b-a > eps*b
    x = (a+b)/2;
    fx = x^3-2*x-5;
    if fx == 0
        break
    elseif sign(fx) == sign(fa)
        a = x; fa = fx;
    else
        b = x; fb = fx;
    end
end
x
```

程序运行结果如下：

```
x =    2.0946
```

(3) return 命令。return 命令用于终止当前命令的执行，并且立即返回上一级调用函数或等待键盘输入命令，可以用来提前结束程序的运行。

【例 A-9】计算矩阵的特征值，当遇到空的矩阵时，必须使用 return 指令来处理。

```
function d = det(A)
%det(A)用来计算矩阵 A 的特征值
if isempty(A)
    d = 1;
    return
else
    ...
end
```

(4) warning 命令。warning 命令用于在程序运行时给出必要的警告信息，这在实际中是非常有必要的。在实际中，因为一些人为因素或其他不可预知的因素可能会使某些数据输入有误，如果编程者在编程时能够考虑到这些因素，并设置相应的警告信息，那么就可以大大降低由数据输入有误而导致程序运行失败的可能性。

- warning('message')：显示警告信息“message”。其中，message 为文本信息。
- warning('message', a1, a2,...)：显示警告信息“message”。其中，message 包含转义字符，且每转义字符的值将被转化为 a1,a2,... 的值。
- s = warning(state, mode)：用于显示警告信息的状态。mode 取值为 on 时，显示其后所有 warning 命令的警告信息；取值为 off 时，不显示其后所有 warning 命令的警告信息；取值为 debug 时，即当遇到一个警告时，启动调试程序。

【例 A-10】编写一个求两矩阵相加之和的程序。

```
function S=ABsum(A,B)
```

```
% ABsum 函数用于实现两个矩阵相加
[m1,n1]=size(A);
[m2,n2]=size(B);
%如果 A、B 中有一个为空矩阵或两矩阵的维数不等，则返回空矩阵，并给出警告信息
if isempty(A)
    warning('A 为空矩阵！');
    S=[];
    return;
elseif isempty(B)
    warning('B 为空矩阵！');
    S=[];
    return;
elseif m1 ~= m2 || n1 ~= n2
    warning('两矩阵的维数不等！');
    S=[];
    return;
else
    for i=1:m1
        for j=1:n1
            S(i,j)=A(i,j)+B(i,j);
        end
    end
end
```

保存文件，在命令窗口输入两个矩阵：

```
>> A=[1 7 8;2 6 7]; B=[2 8;3 9;0 7]; %输入维数不相等矩阵
```

输出为：

```
Warning: 两矩阵的维数不等！
> In ABsum at 15
S =
    []
```

重新输入：

```
>> clear all;
>> A=[1 7 8;2 6 7];B=rand(2,3); %输入维数相等矩阵
>> S=ABsum(A,B)
```

输出为：

```
S =
    1.9572    7.8003    8.4218
    2.4854    6.1419    7.9157
```

重新输入：

```
>> clear all;
>> A=[]; B=[2 8;3 9;0 7]; %输入一矩阵为空矩阵
```





```
>> S=ABsum(A,B)
```

输出为：

```
Warning: A 为空矩阵！
```

```
> In ABsum at 7
```

```
S =
```

```
[]
```

## A.4 MATLAB 的二维绘图

二维图形是将平面坐标上的数据点连接起来的平面图形。可以采用不同的坐标系，除直角坐标系外，还可采用对数坐标、极坐标。数据点可以用向量或矩阵形式给出，类型可以是实型或复型。二维图形的绘制无疑是其他绘图操作的基础。

### 1. 二维绘图

在 MATLAB 中，提供了很多二维绘图函数，其中比较常用且最为基础的是 plot 函数。

#### (1) plot 函数

plot 函数用于绘制 xy 平面上的线性坐标曲线图，因此需要提供一组 x 坐标及其各点对应的 y 坐标，这样就可以绘制分别以 x 与 y 为横、纵坐标的二维曲线。plot 函数的调用格式有以下几种。

- plot(Y)：默认自变量绘图格式，Y 为向量，以 Y 元素值为纵坐标，以元素在 Y 向量中的位置为横坐标绘图。
- plot(X1,Y1,...,Xn,Yn)：当输入参数都为向量时，X1 与 Y1，...，Xn 与 Yn 分别组成一组向量对，每组向量对的长度可以不同。每个向量对都可以绘制出一条曲线，这样可以在同一坐标内绘制出多条曲线。当输入参数为矩阵时，配对的  $X_i$ 、 $Y_i$  按对应列元素为横、纵坐标分别绘制曲线，曲线条数等于矩阵的列数。
- plot(X1,Y1,LineSpec,...,Xn,Yn,LineSpec)：绘制多个同长度向量组 ( $X_i, Y_i$ ) 的图形。如果其中某对 X、Y 是矩阵，则按 X、Y 匹配的方向配对绘制曲线，且每组向量由参数 LineSpec 确定曲线的线型、颜色及标记符号，可以同时使用 2~3 个参数。LineSpec 参数取值见表 A-1。

表 A-1 线型、颜色和标记符号选项

线 型		颜 色		标 识 符 号			
-	实线	b	蓝色	.	点	s	方块符 (Square)
:	虚线	g	绿色	o	圆圈	d	菱形符 (Diamond)
-.	点画线	r	红色	x	叉号	v	朝下三角符号
--	双画线	c	青色	+	加号	^	朝上三角符号
		m	品红色	*	星号	<	朝左三角符号
		y	黄色			>	朝右三角符号
		k	黑色			p	五角星符 (Pentagram)
		w	白色			h	六角星符 (Hexgram)



- `plot(X1,Y1,LineSpec,'PropertyName',PropertyValue)` : 对所有用 `plot` 函数创建的图形进行属性设置。

【例 A-11】对于函数  $y_1 = 3x \sin x^3$  ,  $y_2 = 3(x+1) \cos x^3$  , 在  $[-2, 2]$  区间上绘制它们的曲线。

```
>> clear all;
x=-2:0.01:2;
y1=3*x.*sin(x.^3);
y2=3*(x+1).*cos(x.^2);
plot(x,y1,'r',x,y2,'-');
```

程序运行效果如图 A-2 所示。

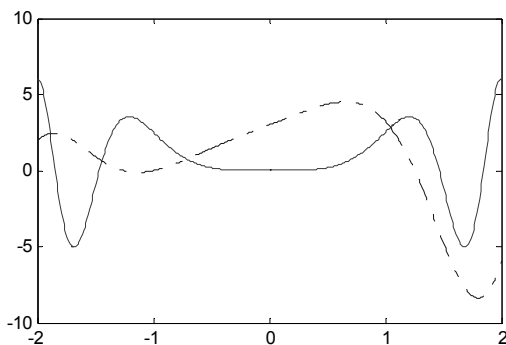


图 A-2 在同一窗口中绘制两条曲线

【例 A-12】在图形窗口中绘制曲线，并改变线型类型、大小及颜色。

```
>> clear all;
x = -pi:pi/10:pi;
y = tan(sin(x)) - sin(tan(x));
plot(x,y,'--rp','LineWidth',2.5,...
      'MarkerEdgeColor','g',...
      'MarkerFaceColor','b',...
      'MarkerSize',12)
```

程序运行效果如图 A-3 所示。

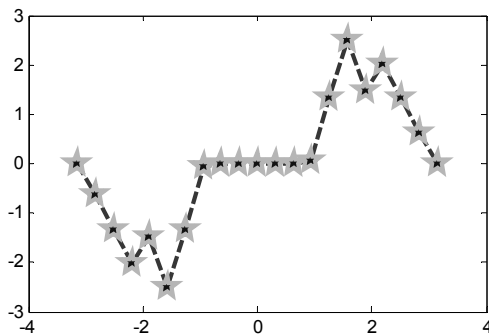
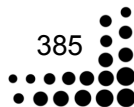


图 A-3 改变曲线线型、大小及标记





## (2) plotyy 函数

在 MATLAB 中,如果需要绘制出具有不同纵坐标标度的两个图形,可以使用 plotyy 函数。这种图形能把函数值具有不同量纲、不同数量级的两个函数绘制在同一坐标中,有利于图形数据的对比分析。plotyy 函数的调用格式如下。

- plotyy(X1,Y1,X2,Y2): 以左、右为同纵轴绘制 X1-Y1、X2-Y2 两条曲线。
- plotyy(X1,Y1,X2,Y2,function): 以左、右不同纵轴把 X1-Y1、X2-Y2 绘制成 function 指定形式的两条曲线。
- plotyy(X1,Y1,X2,Y2,'function1','function2'): 以左、右不同纵轴把 X1-Y1、X2-Y2 绘制成 function1、function2 指定的不同形式的两条曲线。
- [AX,H1,H2] = plotyy(...): 返回 AX 中创建的两个坐标轴的句柄以及 H1 和 H2 中每个图形绘图对象的句柄。AX(1)为左侧轴,AX(2)为右侧轴。

【例 A-13】用不同标度在同一坐标内绘制函数  $y_1 = 3x \sin x^3$  在区间  $[-2,2]$  及  $y_2 = 3(x+1) \cos x^3$  在区间  $[0,4]$  的曲线。

```
>> clear all;  
x1=-2:0.01:2;  
y1=3*x1.*sin(x1.^3);  
x2=0:0.01:4;  
y2=3*(x2+1).*cos(x2.^2);  
[AX,H1,H2]=plotyy(x1,y1,x2,y2);  
set(H1,'LineStyle','--')  
set(H2,'LineStyle',':')
```

程序运行效果如图 A-4 所示。

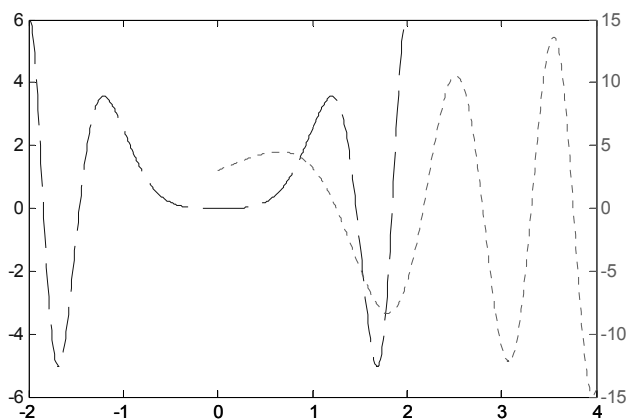


图 A-4 用双纵坐标绘制的曲线

## 2. 图形的辅助工具

绘制完图形后,可能还需要对图形进行一些辅助操作,以使图形意义更加明确,可读性更强。

(1) 图形标注。在默认设置下, MATLAB 给出的图形效果一般都能满足要求,当然用户

也可以根据自己的需要对默认设置做修改。同时, figure 还有很不错的编辑功能, 可以传达更丰富的信息, 这可以通过命令实现, 也可以通过 figure 编辑实现。表 A-2 提供了一些常用的设置选项。

表 A-2 常用的图形标注函数及描述

函 数	描 述	函 数	描 述
title	为图形添加标题	xlabel	x 轴标注
legend	为图形添加图例说明	ylabel	为 y 轴添加标注
text	在指定位置添加文本	colorbar	添加颜色条
grid	是否为图形添加网格线	box	是否为图形添加外框

【例 A-14】为绘制的效果图添加标注说明。

```
>> clear all;
x = -pi:pi/20:pi;
plot(x,cos(x),'-ro',x,sin(x),'-b')
h = legend('cos_x','sin_x',2);
set(h,'Interpreter','none')
xlabel('x'); ylabel('y');
title('绘制两条曲线');
grid on;          %添加网格线
box off;          %删除框图
```

程序运行效果如图 A-5 所示。

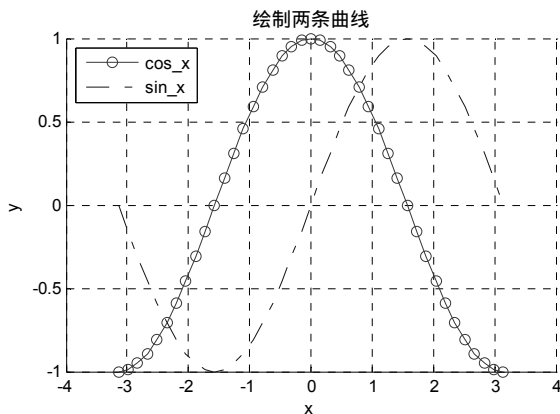


图 A-5 为图形添加标注说明

(2) 多次重叠绘制。在使用 plot 函数时, 有时希望多次绘图, 如果不用命令控制, 则每次运行后, 都会看到当前的图形, 而覆盖上一次的运行结果, 而我们想要的是在一张图纸中多次重叠绘制图形。MATLAB 提供了 hold 函数实现此功能。其调用格式如下。

- hold on: 使当前轴及图形保存而不被刷新, 准备接受此后将绘制的新曲线。
- hold off: 使当前轴及图形不再具备不被刷新的性质。
- hold all: 保留当前的图形及曲线的颜色、风格等, 以保证此后绘制的新曲线的颜色风



格等不会改变。

- hold：当前图形是否具备刷新性质的双向切换开关。
- hold(axes\_handle,...)：利用当前的轴画图。

【例 A-15】利用 hold 绘制离散信号通过零阶保持器产生的波形。

```
>> clear all;  
t=2*pi*(0:20)/20;  
y=cos(t).*exp(-0.5*t);  
stem(t,y,'r');  
hold on;  
stairs(t,y,'r');
```

程序运行效果如图 A-6 所示。

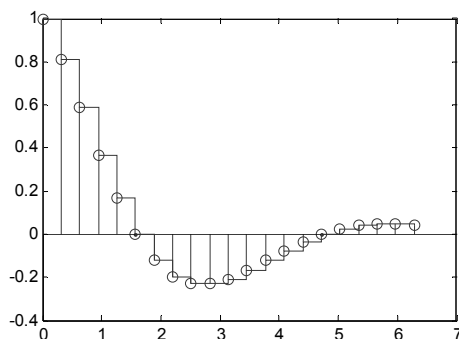


图 A-6 离散信号的重构效果图

(3) 多子图。在实际应用中，经常需要在一个图形窗口内绘制若干个独立的图形，这就需要对图形窗口进行分割。分割后的图形窗口由若干个绘图区组成，每个绘图区都可以建立独立的坐标系并绘制图形。同一图形窗口中的不同图标称为子图。MATLAB 中提供了 subplot 函数，用于将当前图形窗口分割成若干个绘图区。每个区域代表一个独立的子图，也是一个独立的坐标系，可以通过 subplot 函数激活某区，使该区为活动区，所发出的绘图命令都作用于活动区。subplot 函数的调用格式如下。

- subplot(m,n,P)：将产生的图形窗口中有 (m×n) 幅子图，P 是子图的编号。子图的序号编排原则为：左上方为第 1 幅，从左向下依次排号。该调用格式产生的子图分割完全按默认值自动进行。
- subplot(h)：对当前轴进行分割。
- subplot('Position',[left bottom width height])：产生的子图位置由人工指定。指定位置的四元组采用归化的标称单位，即认为图形窗的宽、高的取值范围都为 [0,1]，而左下角为 (0,0) 坐标。
- h = subplot(...)：返回当前分割窗口的句柄值向量 h。

【例 A-16】在一个图形窗口中以子图形式绘制多条曲线。

```
>> clear all;  
x=linspace(0,2*pi,60);  
y=sin(x);
```



```

z=cos(x);
t=sin(x)./(cos(x)+eps);
c=cos(x)./(sin(x)+eps);
subplot(2,2,1);           %选择 2×2 个区中的第 1 号区
stairs(x,y);
title('sin(x)-1');axis([0,2*pi,-1,1]);
subplot(2,1,2);           %选择 2×1 个区中的第 2 号区
stem(x,y);
title('sin(x)-2');axis([0,2*pi,-1,1]);
subplot(4,4,3);           %选择 4×4 个区中的第 3 号区
plot(x,y);
title('sin(x)');axis([0,2*pi,-1,1]);
subplot(4,4,4);           %选择 4×4 个区中的第 4 号区
plot(x,z);
title('cos(x)');axis([0,2*pi,-1,1]);
subplot(4,4,7);           %选择 4×4 个区中的第 7 号区
plot(x,t);
title('tangent(x)');axis([0,2*pi,-40,40]);
subplot(4,4,8);           %选择 4×4 个区中的第 8 号区
plot(x,c);
title('cotangent(x)');axis([0,2*pi,-40,40]);

```

程序运行效果如图 A-7 所示。

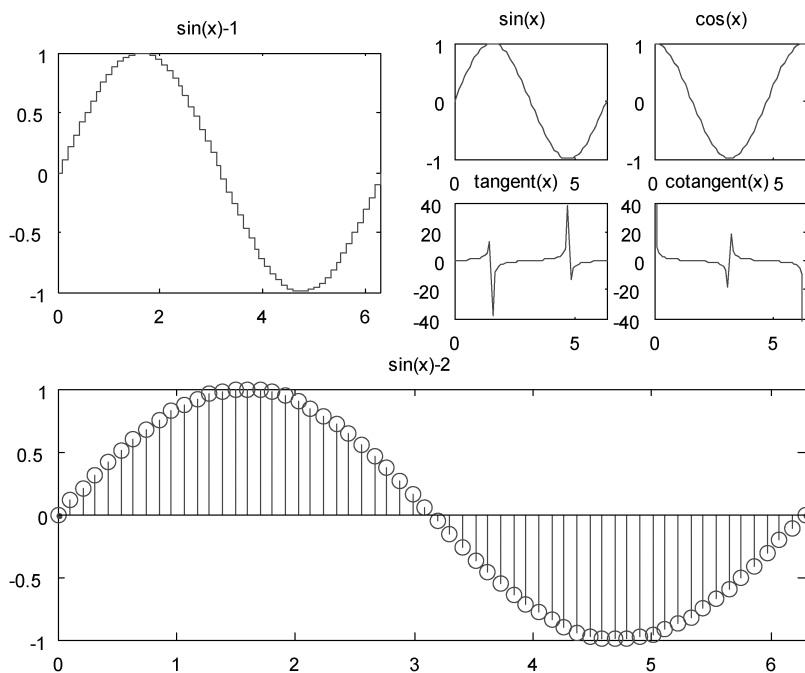
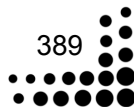


图 A-7 图形窗口的灵活分割





### 3. 其他二维绘图

(1)  $x$ 、 $y$  轴对数坐标。利用 `semilogx`、`semilogy` 函数绘制图形时， $x$ 、 $y$  轴采用对数坐标。若没有指定使用的颜色，则当所绘制的线条较多时，`semilogx`、`semilogy` 函数将自动使用当前轴的 `ColorOrder` 与 `LineStyleOrder` 属性指定的颜色顺序与线型顺序来绘制。其调用格式如下。

- `semilogx(Y)`：对  $x$  轴的刻度求常用对数（以 10 为底），而  $y$  轴为线性刻度。若  $Y$  为实数向量或矩阵，则结合  $Y$  列向量的下标与  $Y$  的列向量绘制出线条；若  $Y$  为复数向量或矩阵，则 `semilogx(Y)` 等价于 `semilogx(real(Y),imag(Y))`。在 `semilogx` 函数的其他调用格式中， $Y$  的虚数部分将被忽略。
- `semilogx(X1,Y1,...)`：结合  $X_i$  与  $Y_i$  绘制出线条，若其中只有  $X_i$  或  $Y_i$  为矩阵，另外一个为向量，行向量的维数等于矩阵的列数，列向量的维数等于矩阵的行数，则按向量的方向分解矩阵，再与向量结合，分别绘制出线条。
- `semilogx(X1,Y1,LineStyle,...)`：按顺序取 3 个参数  $X_i$ 、 $Y_i$ 、`LineStyle` 绘制线条，参数 `LineStyle` 指定使用的线型，标记符号与颜色。用户可以混合使用二参数与三参数形式。
- `semilogx(...,'PropertyName',PropertyValue,...)`：对所有由 `semilogx` 函数生成的图形对象句柄的属性进行设置。
- `h = semilogx(...)`：返回 line 图形句柄向量，每条线对应一个句柄。
- `h = semilogy(...)`：返回 line 图形句柄向量，每条线对应一个句柄，其是相对于  $y$  轴采用对数坐标形式。

(2) 对数坐标。利用 `loglog` 函数绘制图形时， $x$  轴与  $y$  轴均采用对数坐标，其调用格式与 `semilogx` 函数相同。

【例 A-17】在同一窗口不同子图中绘制  $x$  轴对数坐标图形、 $y$  轴对数坐标图形、 $x$  轴与  $y$  轴对数坐标图形。

```
>> clear all;
x=0.001:0.01*pi:2*pi;
y=log10(x);
subplot(2,2,1);plot(x,y);
xlabel('(a)用 plot 函数绘制 log10x 函数曲线');
subplot(2,2,2);semilogx(x,y,'-s')
xlabel('(b)用 semilogx 函数绘制 log10x 函数曲线');
subplot(2,2,3);semilogy(x,y,'-o')
xlabel('(c)用 semilogy 函数绘制 log10x 函数曲线');
subplot(2,2,4);loglog(x,y,'-p')
xlabel('(d)用 loglog 函数绘制 log10x 函数曲线');
```

程序运行效果如图 A-8 所示。

(3) 极坐标图。在极坐标系中绘图，MATLAB 提供了 `polar` 函数来实现。其接受极坐标形式的函数  $\rho=f(\theta)$ ，在笛卡儿坐标系平面上绘制出该函数，且在平面上绘制出极坐标形式的格栅。其调用格式如下。

- `polar(theta,rho)`：用极角  $\theta$  与极径  $\rho$  绘制极坐标图形。极角  $\theta$  为从  $x$  轴到半径的单位为弧度的向量，极径  $\rho$  为各数据点到极点的半径向量。

- `polar(theta,rho,LineStyle)`：参数 `LineStyle` 指定极坐标图中线条的线型、标记符号与颜色等。
- `polar(axes_handle,...)`：利用给定的 `axes_handle` 句柄绘图，并替代之前的轴。
- `h = polar(...)`：返回曲线对象句柄值 `h`。

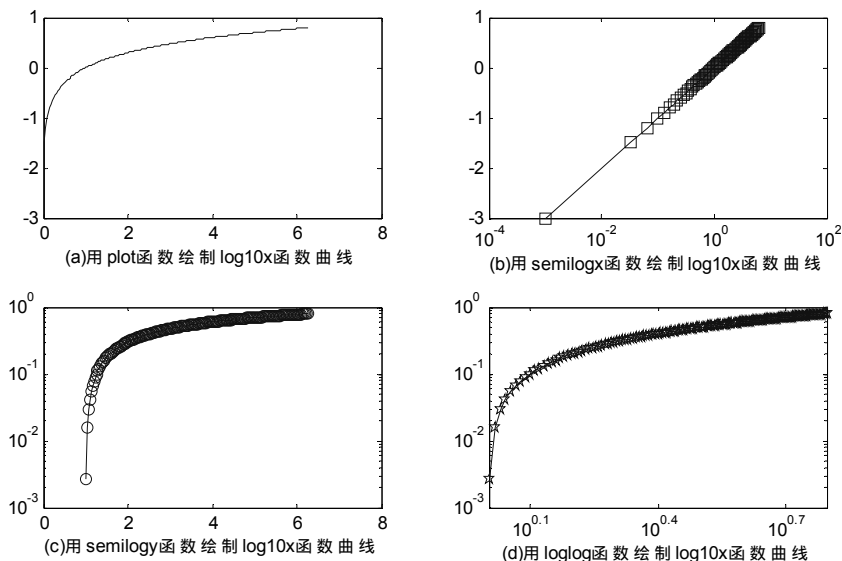


图 A-8 对数坐标效果图

## 【例 A-18】极坐标效果图的绘制。

```
>> clear all;
theta=-pi:0.01:pi;
r=[sin(5*theta).^2;cos(8*theta).^2;sin(theta).^3;cos(5*theta).^3];
for k=1:4
    subplot(2,2,k)
    polar(theta,r(k,:))
end
```

程序运行效果如图 A-9 所示。

(4) 柱坐标。在柱坐标系中绘图，MATLAB 提供了 `pol2cart` 函数来实现。该函数用于将极坐标值或柱坐标值转换成直角坐标系下的坐标值，然后使用 `plot3`、`mesh` 函数绘图，即在直角坐标系下绘制使用柱坐标值描述的图形。其调用格式如下。

- `[X,Y] = pol2cart(THETA,RHO)`：用极角 `theta` 与极径 `rho` 绘制柱坐标图形，并且 `theta` 与 `rho` 的大小相同；`X`、`Y` 分别为返回的 `x` 轴与 `y` 轴数据。
- `[X,Y,Z] = pol2cart(THETA,RHO,Z)`：用极角 `theta`、极径 `rho` 及纯数据 `Z` 绘制柱坐标图形，并且 `theta`、`rho`、`Z` 的大小相同；`X`、`Y`、`Z` 分别为返回的 `x` 轴、`y` 轴及 `z` 轴数据。



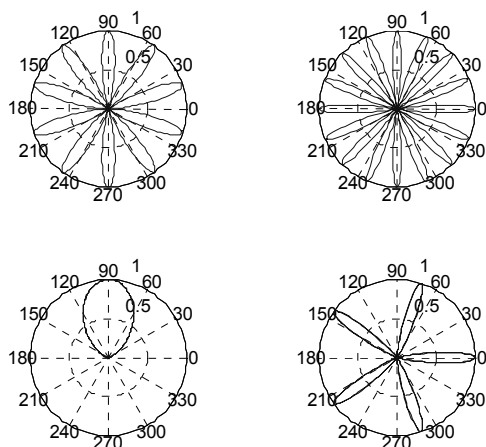


图 A-9 极坐标效果图

【例 A-19】运用 pol2cart 函数绘制渐开线。

```
>> clear all;  
theta=0:pi/20:2*pi;  
rho=sin(theta);  
[X,Y]=meshgrid(theta,rho);  
Z=Y.*X;  
[X,Y,Z]=pol2cart(X,Y,Z);  
mesh(X,Y,Z);  
set(gcf,'color','w'); %设置图形背景颜色为白色
```

程序运行效果如图 A-10 所示。

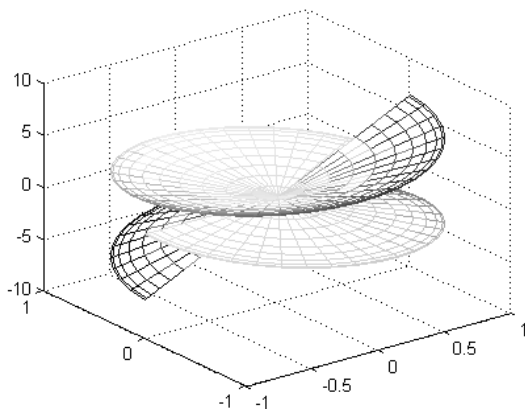


图 A-10 柱坐标效果图

(5) 其他形式的二维图形。有时根据需要可能会用到其他一些图形绘制方法，比如在分析各种成分的比例时，可能需要饼图或条状图，用了 plot 函数作基础，这些方法就很容易掌握。表 A-3 列出了一些常用的二维绘图函数。

表 A-3 常用的二维绘图函数

函 数	描 述	函 数	描 述
rose	玫瑰花图	scatter	散点图
stairs	阶梯图	compass	罗盘图
contour	等高线图	stem	火柴杆图
hist	直方图	pie	饼形图
quiver	向量图	area	面积图
bar、barh	水平与垂直条形图	errorbar	误差图

【例 A-20】利用 contour 函数绘制等高线图。

```
>> clear all;
Z = peaks(100);
figure;
set(gcf,'position',[400,100,600,600], 'color','w')
subplot(2,2,1);
cl = [-7:1:10];          % Define contour levels for all plots
contour(Z, cl)
axis([0 100 0 100]); colormap autumn;
set(gca,'Xtick',[0 100],'Ytick',[0 100]);
title('Peaks Surface (underlying data)')
ZN = Z + rand(100) - .5;
subplot (2,2,2)
contour(ZN, cl)
axis([0 100 0 100]);
set(gca,'Xtick',[0 100],'Ytick',[0 100]);
title('Peaks Surface (noise added)')
F = [.05 .1 .05; .1 .4 .1; .05 .1 .05];
ZC = conv2(ZN,F,'same');
subplot (2,2,3)
contour(ZC, cl)
axis([0 100 0 100]);
set(gca,'Xtick',[0 100],'Ytick',[0 100]);
title('Noisy Surface (smoothed once)')
ZC2 = conv2(ZC,F,'same');
subplot (2,2,4)
contour(ZC2, cl)
axis([0 100 0 100]);
set(gca,'Xtick',[0 100],'Ytick',[0 100]);
title('Noisy Surface (smoothed twice)')
```

程序运行效果如图 A-11 所示。

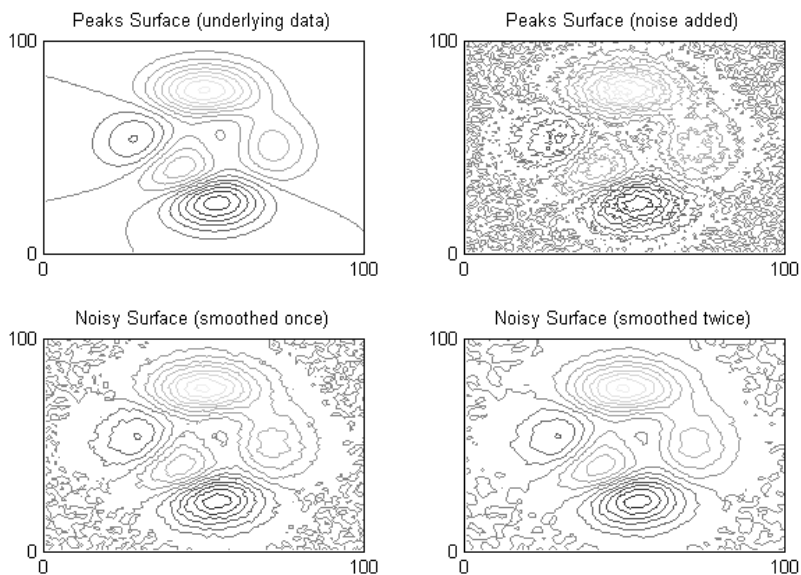


图 A-11 等高线的绘图

【例 A-21】绘制函数  $y = e^{-x^2}$  的条形图。

```
>> clear all;  
x = -2.9:0.2:2.9;  
subplot(1,2,1);bar(x,exp(-x.*x),'r');  
xlabel('(a) 垂直条形图');  
subplot(1,2,2);barh(x,exp(-x.*x),'y');  
xlabel('(b) 水平条形图');
```

程序运行效果如图 A-12 所示。

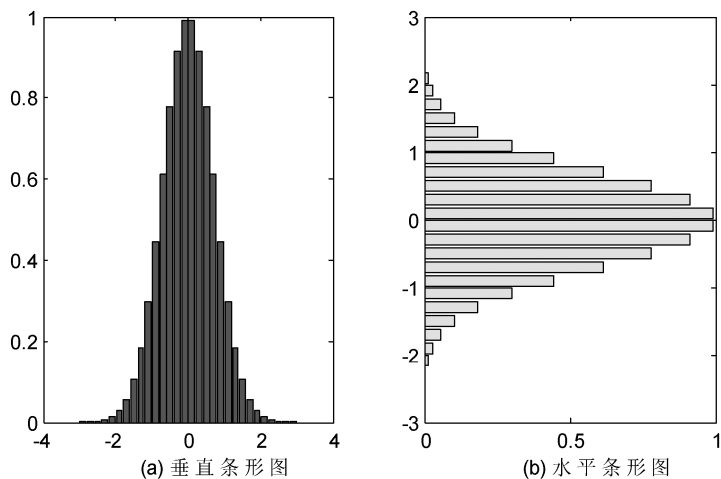


图 A-12 条形图



【例 A-22】绘制正弦波信号的误差图。

```
>> clear all;  
X = 0:pi/10:pi;  
Y = sin(X);  
E = std(Y)*ones(size(X));  
errorbar(X,Y,E)
```

程序运行效果如图 A-13 所示。

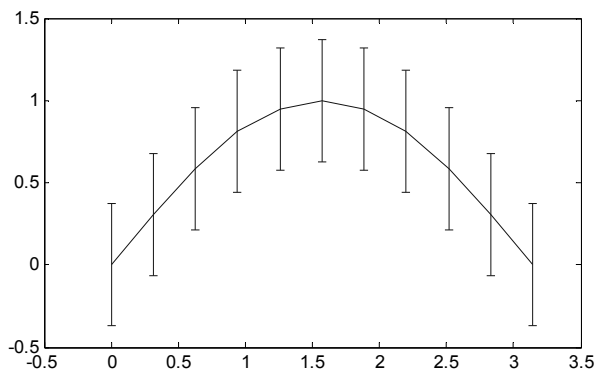


图 A-13 误差图

【例 A-23】在绘制的等高线上绘制向量图。

```
>> clear all;  
n = -2.0:2.0;  
[X,Y,Z] = peaks(n);  
contour(X,Y,Z,10);  
[U,V] = gradient(Z,2);  
hold on  
quiver(X,Y,U,V)  
hold off
```

程序运行效果如图 A-14 所示。

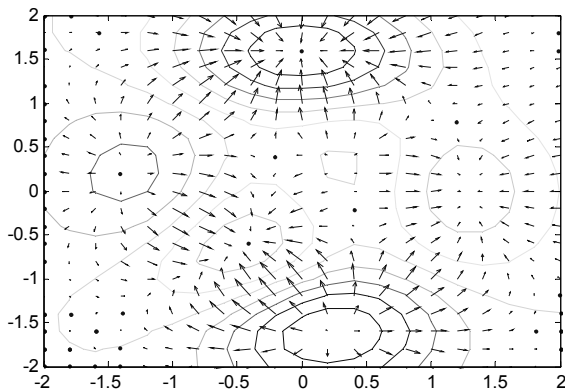
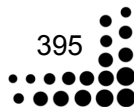


图 A-14 向量图





## 【例 A-24】羽毛图的绘制。

```
>> clear all;  
t = 0:0.5:10;          % 时间限制  
s = 0.05+i;            % 旋转率  
Z = exp(-s*t);          % 计算衰减指数  
feather(Z)              % 羽毛图
```

程序运行效果如图 A-15 所示。

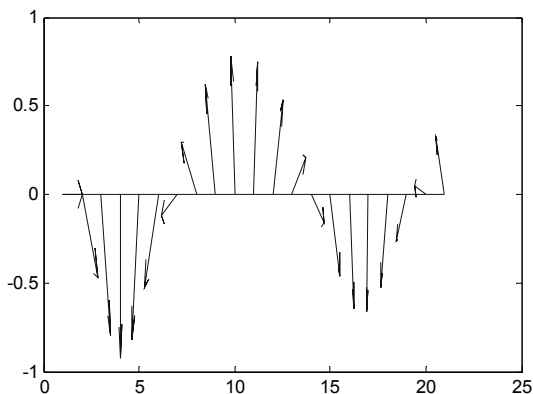


图 A-15 羽毛图

## 【例 A-25】根据给定的数据点绘制饼图。

```
>> clear all;  
X = [19.3 22.1 51.6; 34.2 70.3 82.4; 61.4 82.9 90.8; 50.5 54.9 59.1; 29.4 36.3 47.0];  
x = sum(X);  
explode = zeros(size(x));  
[c,offset] = max(x);  
explode(offset) = 1;          %为 1 时表示扇形分离，为 0 时即不分离  
h = pie(x,explode);  
colormap summer              %添加颜色效果
```

程序运行效果如图 A-16 所示。

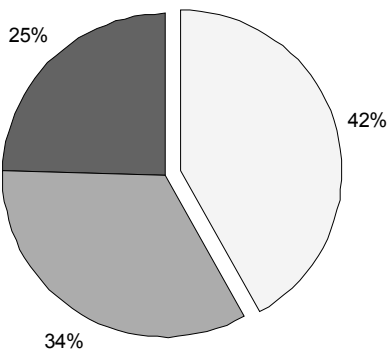


图 A-16 饼图

## 附录 B Fourier 变换与 MATLAB 实现



### B.1 复数形式的 Fourier 级数及其 MATLAB 应用

#### 1. 基本理论

离散 Fourier 变换形式为

$$x_i = \frac{a_0}{2} + \sum_{k=1}^m \left( a_k \cos \frac{2\pi ki}{N} + b_k \sin \frac{2\pi ki}{N} \right)$$

式中,

$$a_0 = \frac{1}{N\Delta t} \sum_{i=0}^{N-1} x_i \Delta t = \frac{2}{N} \sum_{i=0}^{N-1} x_i ; \quad a_k = \frac{1}{N\Delta t} \sum_{i=0}^{N-1} x_i \cos \frac{2\pi ki}{N} \Delta t = \frac{2}{N} \sum_{i=0}^{N-1} x_i \cos \frac{2\pi ki}{N} ;$$

$$b_k = \frac{1}{N\Delta t} \sum_{i=0}^{N-1} x_i \sin \frac{2\pi ki}{N} \Delta t = \frac{2}{N} \sum_{i=0}^{N-1} x_i \sin \frac{2\pi ki}{N} , \quad k = 0, 1, 2, \dots, m。$$

如果将  $a_k$  和  $b_k$  表示为一个复数实部和虚部的某种函数的组合, 并将  $\cos \frac{2\pi ki}{N}$ 、 $\sin \frac{2\pi ki}{N}$  也表示为一个复数的实部和虚部的某种组合, 则可得到

$$x_i = \sum_{k=0}^{N-1} c_k e^{j \frac{2\pi ki}{N}}, \quad i = 0, 1, 2, \dots, N-1 \quad (\text{B-1})$$

$$c_k = \frac{1}{N} \sum_{i=0}^{N-1} x_i e^{-j \frac{2\pi ki}{N}}, \quad k = 0, 1, 2, \dots, N-1 \quad (\text{B-2})$$

注意: 这里的  $c_k$  为复数序列, 其模序列 Fourier 对应于 Fourier 变换所分解的各谐波的振幅; 其频率由  $e^{j \frac{2\pi ki}{N}}$  确定, 即  $\frac{k}{N\Delta t}$ ,  $\Delta t$  为采样间隔。若假定采样间隔为 1s, 则频率为  $\frac{k}{N}$ 。其初相由实部和虚部的相对大小来确定。这样, 我们就将一个信号分解为多种频率的信号。

若无某种频率的信号, 则该频率信号的振幅为零。另外, 考虑到欧拉公式  $e^{-j \frac{2\pi ki}{N}} = \cos(2\pi \frac{k}{N} i) - j \sin(2\pi \frac{k}{N} i)$ , 当  $k$  由  $0 \sim N-1$  变化时, 后半部分复序列是前半部分复序列的共轭。这样, 由式 (C-2) 得到的复序列也具有后半部分复序列是前半部分复序列的共轭的性质。由于后半部分复序列是前半部分复序列的共轭, 因此, 我们将  $k$  取到  $N-1$ , 并没有增加独立参数的个数。



这种形式是 Fourier 变换的最基本的形式,以后所讲的 Fourier 变换都是指这种形式。这样,就建立了复数形式 Fourier 变换之间的关系。已知时间域内的等间隔数据,要求频率域内的振幅,采用式 (B-2),称之为 Fourier 变换;已知频率域内的情况求时间域的值,采用式 (B-1),称之为 Fourier 逆变换。

## 2. Fourier 变换的 MATLAB 实现

【例 B-1】求周期序列  $x(n)$  的离散 Fourier 变换。其中,  $x(n) = (0\ 1\ 2\ 3)$

基本周期为  $N = 4$ , 令  $W_4 = e^{-j\frac{2\pi}{4}} = \cos\frac{\pi}{2} - j\sin\frac{\pi}{2} = -j$ , 则 Fourier 变换公式可写为

$$X(k) = \sum_{n=0}^3 x(n)W_4^{nk}, \quad k = 0, 1, 2, 3$$

将数据代入上式可得

$$X(0) = 6, X(1) = -2 + 2j, X(2) = -2, X(3) = -2 - 2j$$

根据本例的解法,可以将 Fourier 变换写成下面的程序:

```
function [Xk]=dfs(xn,N)
% 计算离散 Fourier 序列的系数
% 调用方式:[Xk]=dfs(xn,N)
% Xk: 离散 Fourier 变换的系数,序号:0<=k<=N-1
% xn: 取周期信号的一个周期,序号:0<=k<=N-1
% N: xn 的数据个数
n=[0:1:N-1];           % 数据数组序列
k=[0:1:N-1];           % 频率数组序列
WN=exp(-j*2*pi/N);      % Wn 因子
nk=n'*k;                % 产生 N×N 的矩阵 nk
WNnk=WN.^nk;            % 离散 Fourier 变换矩阵
Xk=xn*WNnk;             % Fourier 变换系数
```

这里巧妙地利用了 MATLAB 中矩阵相乘的概念使程序简化,即

$$\mathbf{nk} = \begin{pmatrix} 0 \\ 1 \\ 2 \\ 3 \end{pmatrix} \begin{pmatrix} 0, 1, 2, 3 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 \\ 0 & 2 & 4 & 6 \\ 0 & 3 & 6 & 9 \end{pmatrix}$$

$\mathbf{WNnk} = \mathbf{WN}^{\mathbf{nk}}$  也是一个  $4 \times 4$  的矩阵。利用矩阵  $\mathbf{xn} = (0\ 1\ 2\ 3)$  与矩阵  $\mathbf{WN}$  和  $\mathbf{nk}$  相乘中的相加,即相加的运算蕴含在矩阵相乘的运算中,请大家注意 MATLAB 中的这种简便方法。这里有一个问题,即得出的 Fourier 变换与前面所讲的变换差一个系数,所得的结果要乘以  $2/N$  才为某种频率的真实振幅,因此要想得到真实振幅,必须将结果乘以  $2/N$ 。

Fourier 逆变换可以用下列程序表示,其中的分析原理与前面相同:

```
function [xn]=idfs(Xk,N)
% 计算离散 Fourier 逆变换
% 调用方式:[xn]=idfs(Xk,N)
% xn: 输出的 Fourier 逆变换后的序列,序号:0<=n<=N-1
```



```
% Xk: 输入数组, 序号: 0<=n<N-1
% N: Xk 的元素个数
n=[0:1:N-1];           % 数据数组序列
k=[0:1:N-1];           % 频率数组序列
WN=exp(-j*2*pi/N);      % Wn 因子
nk=n'*k;                % 产生 N x N 的矩阵 nk
WNnk=WN.^(-nk);         % 离散 Fourier 变换矩阵
xn=(Xk*WNnk)/N;         % Fourier 变换系数
```

【例 B-2】重新用程序求解例 B-1 的 Fourier 变换。

程序如下：

```
xn=[0 1 2 3];
N=4;
Xk=dfs(xn,N)
```

程序运行结果为：

```
Xk =
    6.0000    -2.0000 + 2.0000i    -2.0000 - 0.0000i    -2.0000 - 2.0000i
```

可以发现，得到的结果与上面的结果一致，这也进一步证明了程序的正确性。

### 3. MATLAB 程序实例

前面我们对 Fourier 变换进行了理论分析，并给出了程序，这里将给出 Fourier 变换实例。

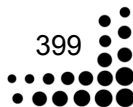
我们已经了解，振幅相同、频率相差不大的两个振动可以合成为一种称为“拍”的、振幅周期性变化的振动。那么从拍的数据中通过 Fourier 变换能否分析出其中的频率成分呢？请看下面的例子。

【例 B-3】已知序列  $x(n) = \cos(2\pi \cdot 0.24n) + \cos(2\pi \cdot 0.26n)$ ， $n = 0 \sim 99$ ，试绘制  $x(n)$  及其 Fourier 变换的幅值图，并用变换后的数值求解逆变换，其中采样周期为 1s。

我们知道，假定采样频率为 1Hz（以 1s 的采样周期进行采样），则该信号包含有 0.24Hz 和 0.26Hz 两种频率的波，其振幅均为 1。

程序如下：

```
clf                                %清除图形框的内容
N=100;dt=1;                       %设置最大点数
n=0:N-1; t=n*dt;                  %给出时间序列
xn=cos(2*pi*0.24*t)+cos(2*pi*0.26*t); %给出原始信号的序列
Xk=dfs(xn,N);                     %对原始信号进行 Fourier 变换
magXk=abs(Xk); phaXk=angle(Xk);   %求出 Fourier 变换的振幅和相位
subplot(2,2,1), plot(t,xn); xlabel('时间/s') %绘出原始信号
title('原始信号(N=100)');
xx=idfs(Xk,N);                    %Fourier 逆变换
x=real(xx);                       %取变换后的实部，如做试验可以验证其虚部为零
subplot(2,2,2), plot(t,x), xlabel('时间/s'), title('运用 Fourier 逆变换得到的合成信号')
k=0:length(magXk)-1;
```







```
subplot(2,1,2),plot(k/(N*dt),magXk*2/N);    %绘出 Fourier 变换的振幅谱
%采用真实振幅绘图
xlabel('频率/Hz'),ylabel('振幅');
title(' X(k)振幅(N=100)');
```

程序的运行结果如图 B-1 所示。程序中的函数  $\text{abs}(x)$  用于计算复向量  $x$  的幅值； $\text{angle}(x)$  用于计算复向量  $x$  的相角，介于  $-\pi \sim \pi$ ，用弧度表示。可以看出，Fourier 变换后确实分析出频率为 0.24Hz 和 0.26Hz 的振动。而其他频率成分的幅值为零，表明信号中不存在其他频率成分。运用逆变换完全恢复了原始信号，表明了该程序的正确性。

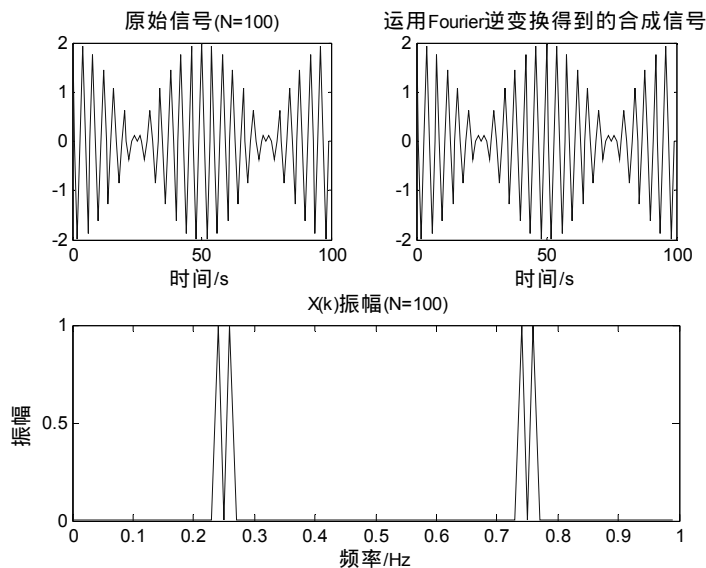


图 B-1 绘制  $x(n)$  及其 Fourier 变换的幅值图

由于前半部分复序列是后半部分复序列的共轭，因此，Fourier 变换后的振幅谱以 0.5Hz 为对称轴将振幅序列分为对称的两部分。

本例中没有给出信号的相位谱。要想得到相位谱，只要在程序中加以适当修改即可。

【例 B-4】对于例 B-3 中的  $x(n)$ ，通过补零增长至 120 个数，绘制  $x(n)$  和它的离散 Fourier 幅值图，将原始信号与逆变换恢复的结果进行比较。

程序如下：

```
clf                                %清除图形框的内容
N=120;dt=1;                       %设置最大点数
n=0:N-1; t=n*dt;                  %给出时间序列
xn=cos(2*pi*0.24*[0:99])+cos(2*pi*0.26*[0:99]);
xn=[xn,zeros(1,N-100)];           %给出原始信号的值序列
Xk=dfs(xn,N);                     %对原始信号进行 Fourier 变换
magXk=abs(Xk);phaXk=angle(Xk);     %求出 Fourier 变换的振幅和相位
subplot(2,2,1),plot(t,xn); xlabel('时间/s') %绘出原始信号
title('原始信号(N=120)');
xx=idfs(Xk,N);                    %Fourier 逆变换
```



```

x=real(xx); %取变换后的实部，如做试验可以验证其虚部为零
subplot(2,2,2),plot(t,x),xlabel('时间/s'),title('运用 Fourier 逆变换得到的合成信号')
k=0:length(magXk)-1;
subplot(2,1,2),plot(k/(N*dt),magXk*2/N); %绘出 Fourier 变换的振幅谱
%采用真实振幅绘图
xlabel('频率/Hz');ylabel('振幅');
title('X(k)振幅(N=120)');

```

程序运行结果如图 B-2 所示。可以看到，例 B-3 中的“拍”信号加上 20 个零后，其 Fourier 变换分解的频率成分除了含有原有的 0.24Hz 和 0.26Hz 频率信号外，还含有很多其他频率的小振幅振动，也就是说，多种频率的小振幅振动叠加才能使用后面的时间域信号为零。

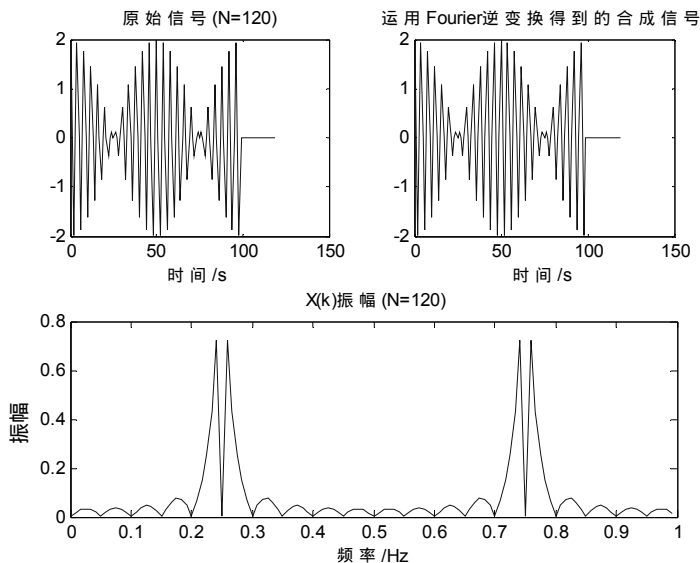


图 B-2 绘制  $x(n)$  补零后及其 Fourier 变换的幅值图

## B.2 Fourier 变换的性质

B.1 介绍了 Fourier 变换的基本理论，为了进一步了解 Fourier 变换，本节对 Fourier 变换的性质进行讲解。

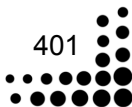
### 1. Fourier 变换的线性性

若  $x(n)$  和  $y(n)$  分别具有离散 Fourier 变换  $X(k)$  和  $Y(k)$ ，则  $x(n) + y(n)$  的 Fourier 变换为  $X(k) + Y(k)$ 。

此定理通过公式  $X(k) = \sum_{n=0}^{N-1} x(n)e^{-j\frac{2\pi}{N}kn}$  ( $k = 0, 1, 2, \dots$ ) 可立即得出。

下面的例子简单说明了 Fourier 变换的线性性。

【例 B-5】设有两个振动  $\cos(2\pi \cdot 0.24t)$  和  $\cos(2\pi \cdot 0.12t)$ ，求两个振动及其合成振动的





Fourier 变换，采样周期为 1s，数据长度为 100。

程序如下：

```
clf
N=100; dt=1; %数据长度为 100，采样周期为 1s
n=0:N-1; t=n*dt; %给出时间序列
xn1=cos(2*pi*0.24*t); %第一个振动
xn2=cos(2*pi*0.12*t); %第二个振动
Xk1=dfs(xn1,N); %第一个振动的 Fourier 变换
Xk2=dfs(xn2,N); %第二个振动的 Fourier 变换
magXk1=abs(Xk1); phaXk1=angle(Xk1); %第一个振动的振幅、相位
magXk2=abs(Xk2); phaXk2=angle(Xk2); %第二个振动的振幅、相位
k=0:length(magXk1)-1;
subplot(3,1,1), plot(k/(N*dt), magXk1*2/N); %绘制第一个振动的振幅谱
ylabel('振幅');
title('第一个振动的 Fourier 变换');
k=0:length(magXk2)-1;
subplot(3,1,2), plot(k/(N*dt), magXk2*2/N); %绘制第二个振动的振幅谱
ylabel('振幅');
title('第二个振动的 Fourier 变换');
Xk=dfs(xn1+xn2,N); %两个振动合成的 Fourier 变换
magXk=abs(Xk); phaXk=angle(Xk); %合成振动的振幅和相位
k=0:length(magXk)-1;
subplot(3,1,3), plot(k/(N*dt), magXk*2/N); %绘制合成振动的振幅和相位
xlabel('频率/Hz'); ylabel('振幅');
title('合成振动的 Fourier 变换');
```

程序运行结果如图 B-3 所示。可以看到，两个振动的 Fourier 变换的叠加在频域上与合成振动的 Fourier 变换是一致的，清楚地表明了 Fourier 变换的线性性。

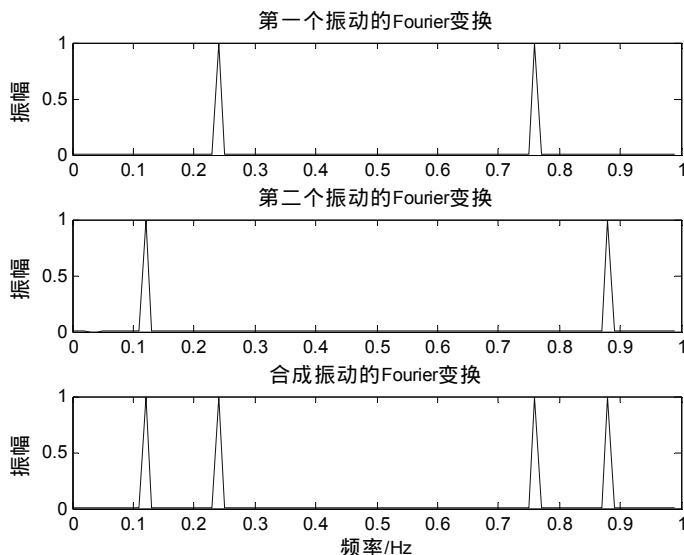


图 B-3 Fourier 变换的线性性



## 2. 尺度变换特性

若  $f(t) \leftrightarrow F(\omega)$ ，则 Fourier 变换的尺度变换特性为

$$f(at) \leftrightarrow \frac{1}{|a|} F\left(\frac{\omega}{a}\right)$$

下面举例说明。

【例 B-6】设  $f(t) = G_2(t) = u(t+1) - u(t-1)$ ，用 MATLAB 求  $y(t) = G_1(t) = u(2t+1) - u(2t-1)$  的频谱  $Y(\omega)$ ，并与  $f(t)$  的频谱  $F(\omega)$  进行比较。

程序如下：

```
R=0.02;t=-2:R:2;
f=heaviside(2*t+1)-heaviside(2*t-1)
W1=2*pi*5;
N=500;k=0:N;W=k*W1/N;
F=f*exp(-j*t'*W)*R;
F=real(F);
W=[-fliplr(W),W(2:501)];
F=[fliplr(F),F(2:501)];
subplot(2,1,1);plot(t,f);
xlabel('t');ylabel('f(t)');
title('f(t)=u(t+1)-u(t-1)');
subplot(2,1,2);plot(W,F);
xlabel('w');ylabel('F(w)');
title('f(t)的 Fourier 变换 F(w)');
```

程序运行结果如图 B-4 所示。

由图 B-4 可见， $Y(\omega)$  将  $F(\omega)$  展宽了一倍，而幅度将为  $F(\omega)$  的一半。

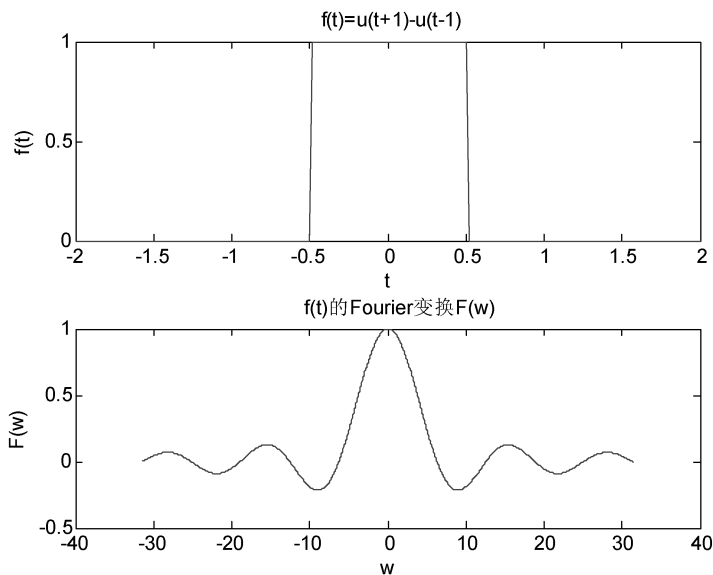
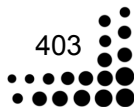


图 B-4 尺度变换的例子





### 3. Fourier 变换时移特性

如果  $x(n)$  的 Fourier 变换为  $X(k)$ ，则将  $x(n)$  移位  $m$ ，即  $x(n-m)$  的 Fourier 变换为  $X(k)e^{-j2\pi km/N}$ 。

该定理证明如下：

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi}{N}kn}, \quad k=0,1,2,\dots \quad (\text{B-3})$$

$$\sum_{n=0}^{N-1} x(n-m) e^{-j\frac{2\pi}{N}k(n-m)} e^{-j\frac{2\pi}{N}km} = X(k) e^{-j\frac{2\pi}{N}km}, \quad k=0,1,2,\dots \quad (\text{B-4})$$

式 (B-4) 左边和右边之所以相等，是因为 Fourier 变换具有循环取值的特点。此定理告诉我们：将信号移位后，即推迟或提前几个时间单位并不会影响 Fourier 变换的振幅，但要影响其相位。

在 MATLAB 中，信号的时移可通过下面的函数实现：

```
function y=cirshfft(n,m,N)
% 对 N 个数据 x 循环移位 m 个单位采样
% 调用形式：[y]=cirshfft(x,m,N)
% y：循环移位后的输出序列
% x：输入序列
% m：移位的采样单位数
% N：输入序列的大小
if length(x)>N
    error('N must be>=the length of x')
end
x=[x zeros(1,N-length(x))];
n=[0:1:N-1];
n=mod(n-m,N);
y=x(n+1);
```

【例 B-7】将  $xn1=\cos(0.14\pi n)$  的信号时移 20 个单位，求其 Fourier 变换并将其除以  $e^{-j2\pi k 20/N}$  与时移前信号的 Fourier 变换做比较。采样周期为 1s，数据长度为 128。

程序如下：

```
clf
N=128;dt=1; %数据长度和采样间隔
n=0:N-1; t=n*dt; %时间序列
xn1=cos(0.14*pi*t); %原始信号
subplot(3,2,1),plot(t,xn1);title('原始信号'); %绘制原始信号
xn2=cirshfft(xn1,20,N); %循环移位 20 个时间单位
subplot(3,2,2),plot(t,xn2); % 绘出循环移位后的信号
title('时移 20 个单位的信号')
Xk1=dfs(xn1,N); %将原信号进行 Fourier 变换
magXk1=abs(Xk1);phaXk1=angle(Xk1); %得到原信号的振幅和相位
k=0:length(magXk1)-1;
```



```

subplot(3,2,3),
plot(k/(N*dt),magXk1*2/N);           %绘制原信号的振幅谱
ylabel('振幅');
title('原始信号的振幅谱');
subplot(3,2,4),plot(k/(N*dt),unwrap(phaXk1)),ylabel('相位角/rad')
%绘制原信号的相位谱,unwrap 为将信号解卷绕,即将相位角展开
title('原始信号的相位谱')
Xk2=dfs(xn2,N);                       %绘制移位后信号的振幅谱
Xk2=Xk2./exp(-j*2*pi*k*20/N);         %将移位后的 Fourier 变换与相乘
%如果没有此语句,得出的相位谱会发生变化
magXk2=abs(Xk2);phaXk2=angle(Xk2);
k=0:length(magXk2)-1;
subplot(3,2,5),
plot(k/(N*dt),magXk2*2/N);
xlabel('频率/Hz');ylabel('振幅');
title('移位后与 exp(-j*2*pi*k*20/N)相乘的振幅谱');
subplot(3,2,6),plot(k/(N*dt),unwrap(phaXk2)),ylabel('相位角/rad')
xlabel('频率/Hz')
title('移位后与 exp(-j*2*pi*k*20/N)相乘的相位谱')

```

程序运行结果如图 B-5 所示。程序中函数 `unwrap(p)` 用于展开弧度相位角  $p$ , 即将不在  $-\pi \sim \pi$  的相位角归算到该区间内, 以便于比较。本例表明, 将信号进行移位  $n$  个单位, 可用频域内除以  $e^{-j2\pi kn/N}$  来代替。这与前面的时移位定理的分析是一致的。

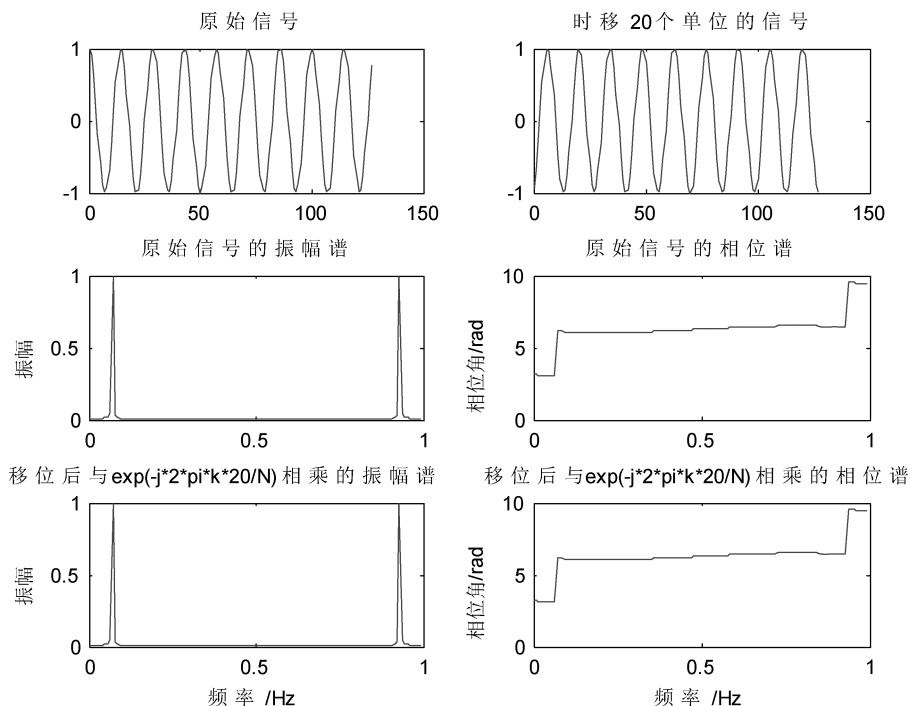
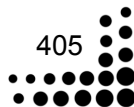


图 B-5 Fourier 变换时移特性





【例 B-8】设  $f(t) = \frac{1}{2}e^{-2t}u(t)$  ,  $y(t) = \frac{1}{2}e^{-2(t-0.3)}u(t-0.3)$  , 试用 MATLAB 绘出  $f(t)$ 、 $y(t)$

及其频谱 ( 幅度谱和相位谱 ) , 并对二者频谱进行比较。

求解  $f(t)$  程序如下 :

```
r=0.02;
t=-5:r:5;
N=200;
W=2*pi*1;
k=-N:N;
w=k*W/N;
f1=1/2*exp(-2*t).*heaviside(t);
F=r*f1*exp(-j*t*w);
F1=abs(F);
P1=angle(F);
subplot(311);
plot(t,f1);
grid;
xlabel('t');
ylabel('f(t)');
title('f(t)');
subplot(312);
plot(w,F1);
xlabel('w');
grid;
ylabel('F(jw)');
subplot(313);
plot(w,P1*180/pi);
grid;
xlabel('w');
ylabel('P(度)');
```

程序运行结果如图 B-6 所示。

将求解  $f(t)$  频谱的程序进行适当修改, 即可得到求解  $y(t)$  频谱的程序, 即将  $t=-5:r:5$  修改为  $t=-2:r:2$ ; 将  $f1$  修改为  $f1=1/2*\exp(-2*(t-0.3))*\text{heaviside}(t-0.3)$ ; 将  $\text{ylabel}('f(t)')$  修改为  $\text{ylabel}('y(t)')$ ; 将  $\text{title}('f(t)')$  修改为  $\text{title}('y(t)')$ 。修改后的程序运行结果如图 B-7 所示。

通过比较图 B-6 和图 B-7 可得, 当时域波形右移后幅度谱不变, 相位增加  $-0.3\omega$ 。

#### 4. Fourier 变换频移特性

如果  $x(n)$  的 Fourier 变换为  $X(k)$ , 则将  $X(k)$  移位  $i$ , 即  $X(K-i)$  的 Fourier 逆变换为  $x(n)e^{j2\pi ki/N}$ 。

将  $r = k - i$  代入 Fourier 变换公式中, 得到



$$X(r) = \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi}{N}rn}, \quad k = 0, \pm 1, \dots$$

则

$$X(k-i) = \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi}{N}(k-i)n} = \sum_{n=0}^{N-1} x(n) e^{j\frac{2\pi}{N}in} x(n) e^{-j\frac{2\pi}{N}kn}$$

由上式可看出,  $x(n)e^{j2\pi ki/N}$  相当于 Fourier 变换的原函数, 即  $X(k-i)$  的 Fourier 变换为  $x(n)e^{j2\pi ki/N}$ 。

此定理得证。此定理还可由下列例子说明。

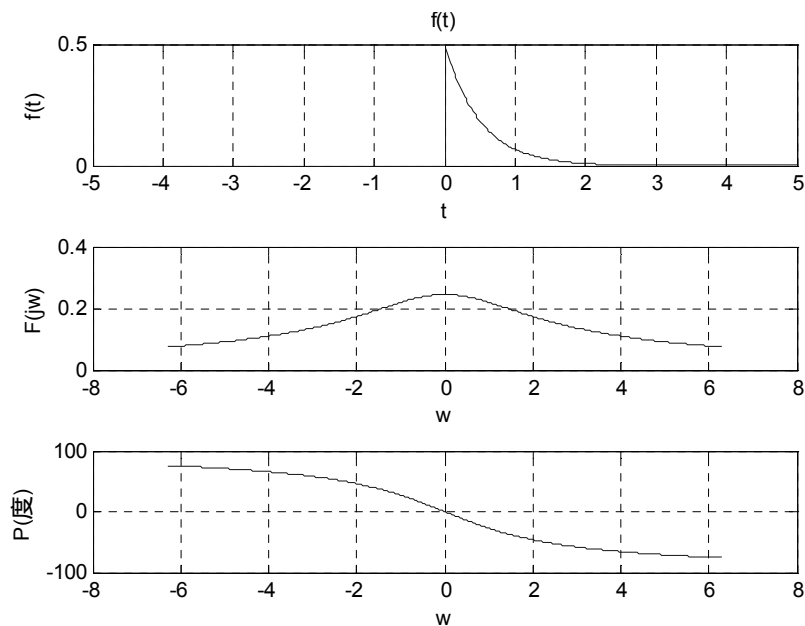


图 B-6  $f(t)$  的频谱图

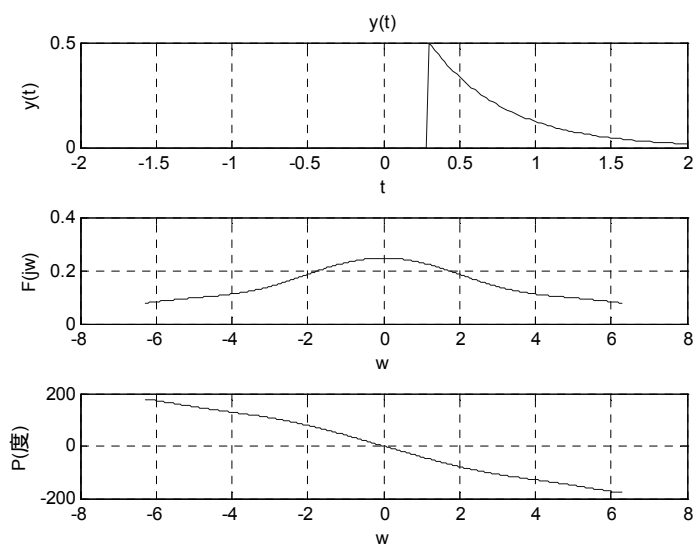
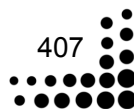


图 B-7  $y(t)=f(t-0.3)$  的频谱图







【例 B-9】设有一序列信号  $x(n) = \cos \frac{\pi n dt}{2}$ ,  $0 \leq n \leq 99$ , 试求得其 Fourier 变换  $X(k)$ , 将其进行循环移位 10 个单位后求解逆变换。比较逆变换的实部和虚部与原始信号和  $x(n)e^{j2\pi k 10/N}$  乘积的实部和虚部对应关系。采样周期  $dt = 1$ 。

程序如下：

```

N=100;dt=1;                                %数据个数与采样间隔
n = 0:N-1;t=n*dt;                            %时间序列值
x = cos(pi*t/2);                             %正弦信号
k = 0:N-1; f = k/(N*dt);                    %频率序列值
X = dfs(x,N);                                %求解正弦信号的 Fourier 变换
realX=real(X);                               %取 X 的实部
imagX=imag(X);                              %取 X 的虚部
Xrealshft=cirshftt(realX,10,N);             %将 X 的实部信号移位 10
Ximagshft=cirshftt(imagX,10,N);             %将 X 的虚部信号移位 10
Y=idfs(Xrealshft+j*Ximagshft,N);           %将移位后信号结合为复数进行逆变换
subplot(2,2,1),plot(t,real(Y));ylim([-1 1]) %Y 的实部
xlabel('时间/s')
title('移位后信号频率域的实部')
subplot(2,2,2),plot(t,imag(Y));             %绘出 Y 的虚部
xlabel('时间/s')
title('移位后信号频率域的虚部')
subplot(2,2,3),plot(t,x.*real(exp(j*2*pi*n*10/N))); %绘 x(n) 的实部
xlabel('时间/s')
title('x*exp(j*2*pi*n*10/N)的 Fourier 变换的实部')
subplot(2,2,4),plot(t,x.*imag(exp(j*2*pi*n*10/N))); %绘 x(n) 的虚部
xlabel('时间/s')
title('x*exp(j*2*pi*n*10/N) 的 Fourier 变换的虚部')

```

程序运行结果如图 B-8 所示。可以看出，将余弦信号进行 Fourier 变换，移位后再进行 Fourier 逆变换得到的结果与原余弦信号 Fourier 变换后和  $e^{j2\pi k 10/N}$  乘积是一致的，从而验证了频移定理。

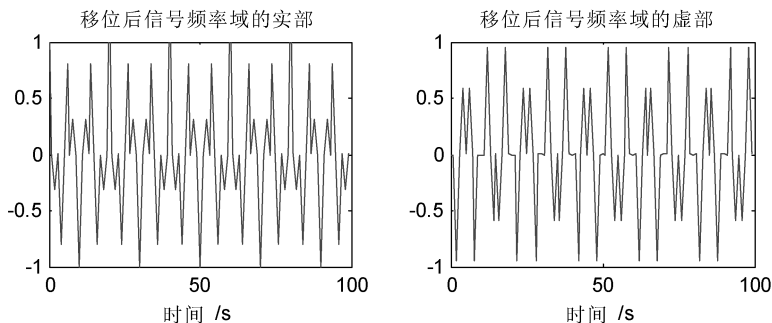


图 B-8 余弦信号验证 Fourier 变换的频移特性

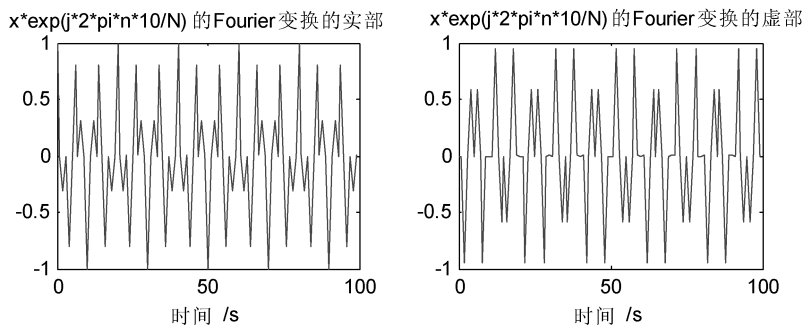


图 B-8 余弦信号验证 Fourier 变换的频移特性 (续)

再举一例说明 Fourier 变换的频移特性。

【例 B-10】设  $f(t) = u(t+1) - u(t-1)$ ，试用 MATLAB 绘出  $f_1(t) = f(t)e^{-j20t}$  及  $f_2(t) = f(t)e^{j20t}$  的频谱  $F_1(\omega)$  和  $F_2(\omega)$ ，并与  $f(t)$  的频谱  $F(\omega)$  进行比较。

程序如下：

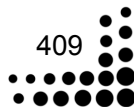
```
R=0.02;t=-2:R:2;
f=heaviside(t+1)-heaviside(t-1);
f1=f.*exp(-j*20*t);
f2=f.*exp(j*20*t);
W1=2*pi*5;
N=500;k=-N:N;W=k*W1/N;
F1=f1.*exp(-j*t*W)*R;
F2=f2.*exp(-j*t*W)*R;
F1=real(F1);
F2=real(F2);
subplot(121);
plot(W,F1);
xlabel('w');
ylabel('F1(jw)');
title('F(w)左移到 w=20 处的频谱 F1(jw)');
subplot(122);
plot(W,F2);
xlabel('w');
ylabel('F2(jw)');
title('F(w)右移到 w=20 处的频谱 F2(jw)');
```

程序运行结果如图 B-9 所示。

## 5. Fourier 变换的对称性

若  $f(t) \leftrightarrow F(\omega)$ ，则 Fourier 变换的对称性为

$$F(t) \leftrightarrow 2\pi f(-\omega)$$



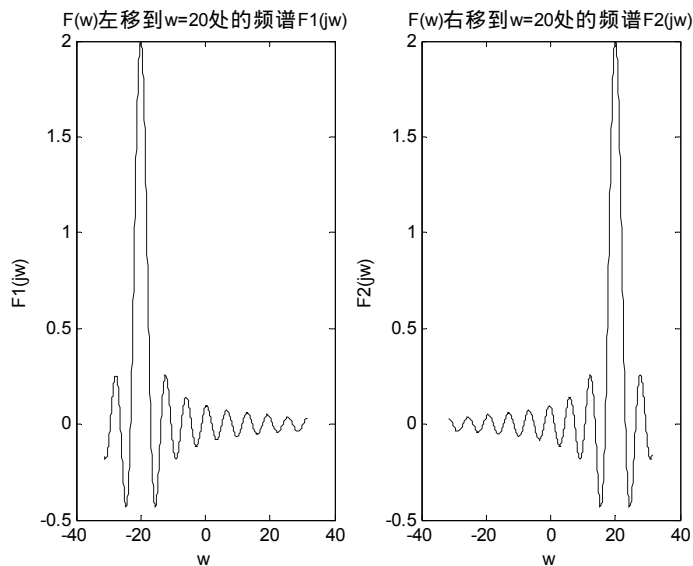


图 B-9 Fourier 变换的频移特性

【例 B-11】设  $f(t) = Sa(t)$ ，已知信号  $f(t)$  的 Fourier 变换为  $F(\omega) = \pi G_2(\omega)$ ，利用 MATLAB 求  $f_1(t) = \pi G_2(t)$  的 Fourier 变换  $F_1(\omega)$ ，并验证对称性。

程序如下：

```
r=0.01;
t=-15:r:15;
f=sin(t)./t;
f1=pi*(heaviside(t+1)-heaviside(t-1));
N=500;
W=5*pi*1;
k=-N:N;
w=k*W/N;
F=r*sinc(t/pi)*exp(-j*t*w);
F1=r*f1*exp(-j*t*w);
subplot(221);
plot(t,f);
xlabel('t');
ylabel('f(t)');
subplot(222);
plot(w,F);
axis([-2 2 -1 4]);
xlabel('w');
ylabel('F(w)');
subplot(223);
plot(t,f1);
axis([-2 2 -1 4]);
xlabel('t');
ylabel('f1(t)');
```



```
subplot(224);
plot(w,F1);
axis([-20 20 -3 7]);
xlabel('w');
ylabel('F1(w)');
```

程序运行结果如图 B-10 所示。

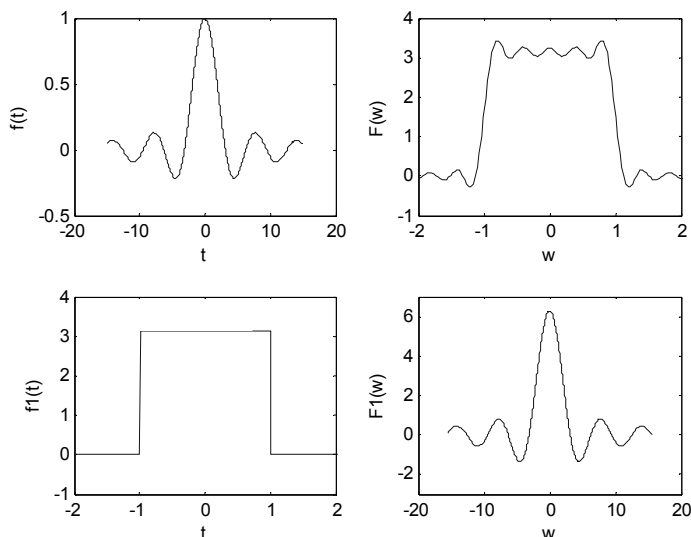


图 B-10 Fourier 变换的对称性

## 6. 偶函数和奇函数与 Fourier 变换后实部和虚部的关系

偶函数的 Fourier 变换对应于频率域的实部，奇函数的 Fourier 变换对应于频率域的虚部。Fourier 变换的这个性质可通过下面的例子进行说明。

**【例 B-12】**对信号  $x = \sin(2\pi \times 0.1t) + 0.5\cos(2\pi \times 0.2t)$ ,  $t = ndt$ ,  $0 \leq n \leq 99$ ,  $dt = 1$ 。求出信号频谱的实部和虚部，同时求出此序列中奇偶部分的 Fourier 变换，与将整个信号进行 Fourier 变换结果的实部和虚部进行比较。

信号中的  $\sin(2\pi \times 0.1t)$  为奇函数， $0.5\cos(2\pi \times 0.2t)$  为偶函数。

程序如下：

```
N=100;dt=1; %数据点数及采样间隔
n = 0:N-1; t=n*dt; %时间序列
x = sin(2*pi*0.1*t)+0.5*cos(2*pi*0.2*t); %数据信号
xe=0.5*cos(2*pi*0.2*t); %信号中的偶函数部分
xo=sin(2*pi*0.1*t); %信号中的奇函数部分
k = 0:N-1; f=k/(N*dt); %频率序列
X = dfs(x,N); %对信号进行 Fourier 变换
XE = dfs(xe,N); %对信号偶函数部分进行 Fourier 变换
XO = dfs(xo,N); %对信号奇函数部分进行 Fourier 变换
XR=real(X); %提取信号 Fourier 变换的实部
```





```

XI=imag(X); %提取信号 Fourier 变换的虚部
subplot(2,2,1); plot(f,XR*2/N); grid on; %绘出信号 Fourier 变换的实部
xlabel('频率/Hz'); ylabel('Re(X)');
title('信号 Fourier 变换的实部')
subplot(2,2,2); plot(f,XI*2/N); grid on; %绘出信号 Fourier 变换的虚部
xlabel('频率/Hz'); ylabel('Im(X)');
title('信号 Fourier 变换的虚部')
subplot(2,2,3); plot(f,real(XE)*2/N); grid on; %绘出信号偶函数部分 Fourier 变换的实部
xlabel('频率/Hz'); ylabel('XE');
title('信号偶函数部分的 Fourier 变换')
subplot(2,2,4); plot(f,imag(XO)*2/N); grid on; %绘出信号奇函数部分 Fourier 变换的虚部
xlabel('频率/Hz'); ylabel('XO');
title('信号奇函数部分的 Fourier 变换')

```

程序运行结果如图 B-11 所示。可以看到，信号 Fourier 变换的实部与信号中的偶函数部分 Fourier 变换的实部完全一致；信号 Fourier 变换的虚部与信号中的奇函数部分 Fourier 变换的结果完全一致。其实，信号偶函数部分 Fourier 变换的虚部以及奇函数部分 Fourier 变换的实部均几乎为零。这就验证了前面结论的正确性。大家可以采用其他函数进行验证。

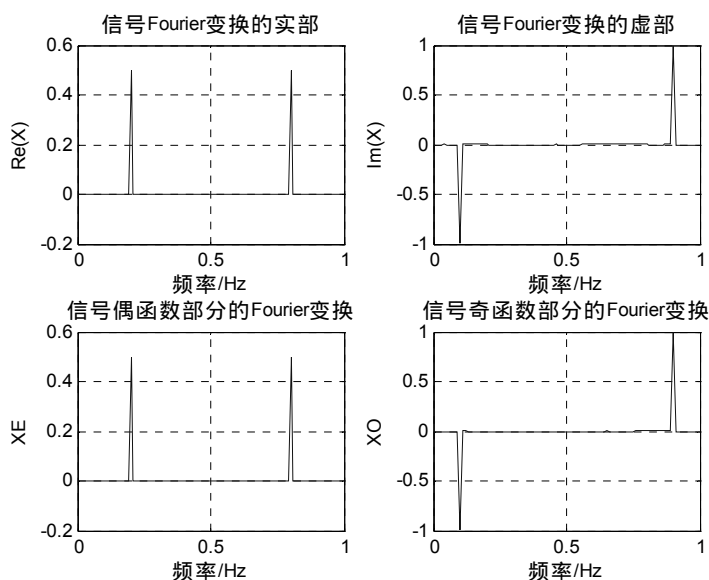


图 B-11 例 B-12 程序运行结果

## 7. 卷积定理

卷积定理可以表述为：两个函数  $h(n)$  和  $x(n)$  在时间域内的卷积等于在频率域内的乘积，

即  $\sum_{i=0}^{N-1} x(i)h(k-i)$  的 Fourier 变换为  $X(k)H(k)$ 。

MATLAB 提供了一个内部函数 `conv`，该函数假定这两个序列都从  $n=0$  开始，利用



$y = \text{conv}(x, h)$  就可实现  $x(n)$  信号和  $h(n)$  信号的卷积。若向量  $x$  的长度为  $n_x$ ，向量  $h$  的长度为  $n_h$ 。则输出  $y$  的向量长度为  $n_y = n_x + n_h - 1$ 。

【例 B-13】产生 2~10Hz 渐变鸟鸣信号，并与脉冲信号进行卷积，将卷积后的 Fourier 变换与这两个函数 Fourier 变换后的乘积进行比较。采样周期为 0.01s，数据长度为 100。

程序如下：

```
N=100; dt=0.01;           %数据点数和采样周期
n=0:N-1; t=n*dt;          %时间序列
f=n/(N*dt);                %频率序列
h=[1 zeros(1,N-1)];        %脉冲信号第一个值为 1，其余为 0
fo=2; fl=10;               %频率渐增函数鸟鸣信号从 2Hz 增加到 10Hz
x=chirp(t,fo,1,fl);         %在时间序列 t 上产生频率渐增信号——鸟鸣信号
xh=conv(x,h);               %将鸟鸣信号和脉冲信号进行卷积，参看上面的用法
XH=dfs(xh(1:N),N);          %由于卷积后数据变长，这里只选取与原数据长度相等的数据个数
subplot(2,2,1), plot(f, real(XH)*2/N);          %绘出卷积后 Fourier 变换的实部
xlabel('频率/Hz'); title('信号卷积后 Fourier 变换 XH 的实部');
subplot(2,2,2), plot(f, imag(XH)*2/N);          %绘出卷积后 Fourier 变换的虚部
xlabel('频率/Hz'); title('信号卷积后 Fourier 变换 XH 的虚部');
X=dfs(x,N);                 %对 chirp 信号进行 Fourier 变换
H=dfs(h,N);                  %对脉冲信号进行 Fourier 变换
XH1=X.*H;                   %对鸟鸣信号和脉冲信号的 Fourier 变换相乘
subplot(2,2,3), plot(f, real(XH1)*2/N);          %绘出 Fourier 变换后乘积的实部
xlabel('频率/Hz'); title(' 频率域乘积 XH1 的实部');
subplot(2,2,4), plot(f, imag(XH1)*2/N);          %绘出 Fourier 变换后乘积的虚部
xlabel('频率/Hz'); title(' 频率域乘积 XH1 的虚部');
```

程序运行结果如图 B-12 所示，可以看出，鸟鸣信号与脉冲信号卷积后的 Fourier 变换与这两个信号 Fourier 变换后的乘积完全相等，验证了卷积定理。

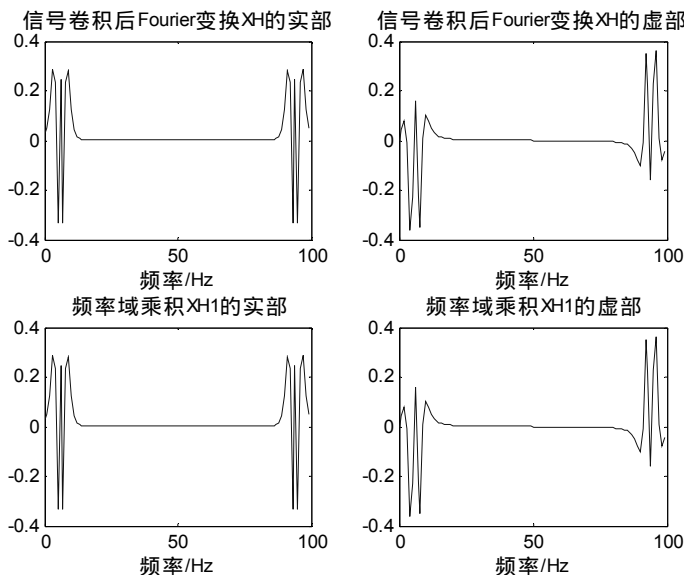
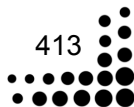


图 B-12 验证卷积定理正确性的实例





【例 B-14】设  $f(t) = u(t+1) - u(t-1)$  ,  $y(t) = f(t) \cdot f(t)$  , 试用 MATLAB 绘出  $f(t)$ 、 $y(t)$ 、 $F(\omega)$ 、 $F(\omega) \cdot F(\omega)$  及  $Y(\omega)$  , 验证卷积定理。

程序如下：

```
R=0.05;t=-2:R:2;
f=heaviside(t+1)-heaviside(t-1);
subplot(321)
plot(t,f)
xlabel('t');
ylabel('f(t)');
y=R*conv(f,f);
n=-4:R:4;
subplot(322);
plot(n,y);
xlabel('t');
ylabel('y(t)=f(t)·f(t)');
axis([-3 3 -1 3]);
W1=2*pi*5;
N=200;
k=N:N;
W=k*W1/N;
F=f*exp(-j*t*W)*R;
F=real(F);
Y=y*exp(-j*n*W)*R;
Y=real(Y);
F1=F.*F
subplot(323);
plot(W,F);
xlabel('w');
ylabel('F(jw)');
subplot(324);
plot(W,F1);
xlabel('w');
ylabel('F(jw)·F(jw)');
axis([-20 20 0 4]);
subplot(325);
plot(W,Y);
xlabel('w');
ylabel('Y(jw)');
axis([-20 20 0 4]);
```

程序运行结果如图 B-13 所示。

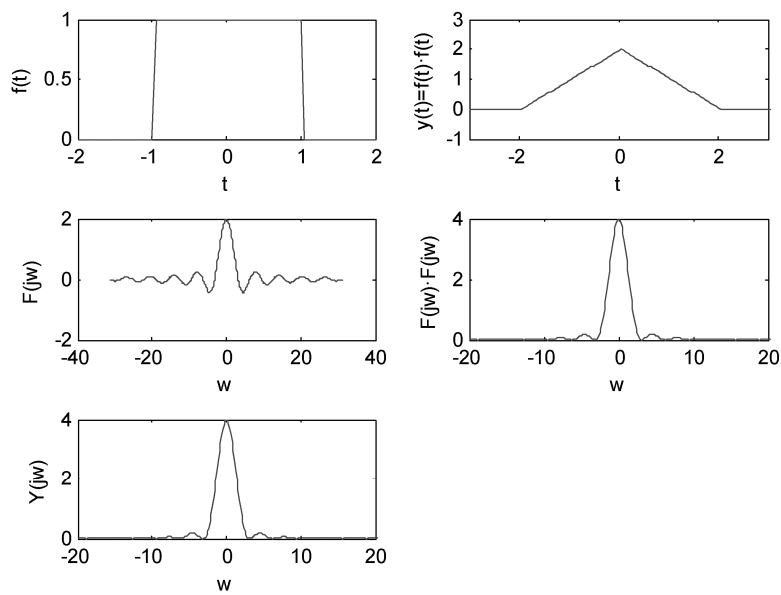


图 B-13 验证卷积定理正确性的实例



## 附录 C Fourier 变换分析与实现

在传统的 Fourier 分析中，信号完全是在频域展开的，不包含任何时域信息，这对于某些应用来说是很恰当的，因为信号频率的信息对其是非常重要的。但其丢弃的时域信息可能对某些应用同样非常重要，所以人们对 Fourier 分析进行了推广，提出了很多能表征时域和频域信息的信号分析方法，如短时 Fourier 变换、Gabor 变换、时频分析、小波变换等。考虑到小波变换域与 Fourier 变换域之间存在着一定的转换关系，并且经典小波分析是从 Fourier 分析的基础上发展出来的，所以首先讲解 Fourier 变换基本理论及其在 MATLAB 中的实现，以便在后续比较小波变换与 Fourier 变换的各自特点和处理问题的不同之处。

### C.1 Fourier 级数与 Fourier 变换

在科学试验与工程技术中经常碰到一种周期运动，比如单摆在振幅很小时的摆动，交流电的电流、电压等，都可用正弦函数  $y = A \sin(\omega x + \varphi)$  来描述，这种运动也常称简谐运动。若干个简谐运动

$$y_k = A_k \sin(k\omega x + \varphi_k), \quad k = 1, 2, \dots, n$$

的叠加可描述更复杂的周期运动。比如，如图 C-1 所示方波就可看成无穷多个奇次正弦函数的叠加。

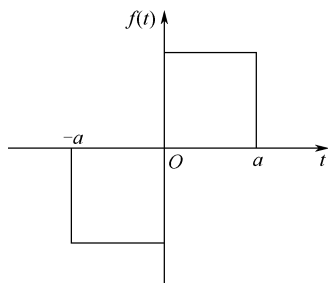


图 C-1 方波示意图

```
clf, x=-pi:pi/100:pi;
y1=sin(x)+1./3*sin(3*x);
y2=y1+1/5*sin(5*x);
y3=y2+1/7*sin(7*x)+1/9*sin(9*x)+1/11*sin(11*x)+1/13*sin(13*x);
plot(x,y1,'-',x,y2,'+',x,y3)
legend('y1','y2','y3')
```

程序运行结果如图 C-2 所示。

形如  $\sum_{k=1}^{\infty} A_k \sin(kx + \varphi_k)$  的函数项级数称为三角级数。三角级数的理论在函数概念中起到很大作用，还有许多数学上的基本概念也来自三角级数，这一理论一直是现代数学的一个重要分支。

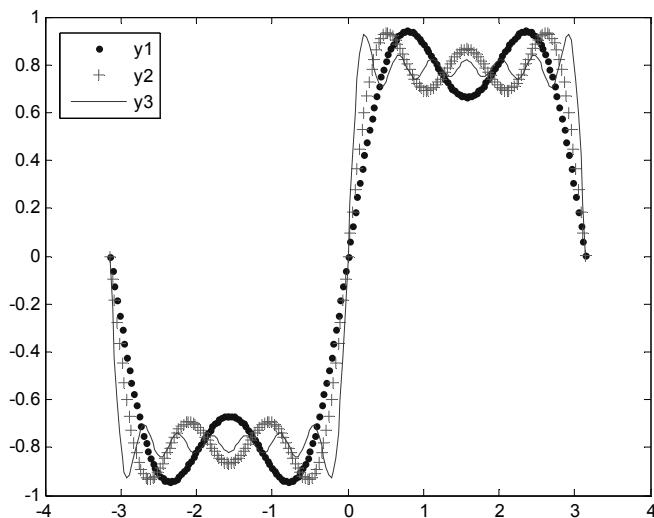


图 C-2 多个奇次正弦函数的叠加

## C.2 三角级数

将  $A_n \sin(n\omega + \varphi_n)$  展开, 并取  $\omega = 1$ , 则三角级数可化成如下形式。

$$\frac{a_0}{2} + \sum_{n=1}^{\infty} (a_n \cos nx + b_n \sin nx) \quad (\text{C-1})$$

我们知道, 三维空间向量可表示为

$$\vec{r} = x\vec{i} + y\vec{j} + z\vec{k} \quad (\text{C-2})$$

其中,  $x$ 、 $y$ 、 $z$  分别为向量  $\vec{r}$  在三个坐标轴上的(投影)分量。

若三角级数 (C-1) 收敛, 并记

$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty} (a_n \cos nx + b_n \sin nx) \quad (\text{C-3})$$

将式 (C-3) 与 (C-2) 中  $\vec{r}$  的表示式对照可以发现, 三角级数其实就是  $f(x)$  在无穷维(函数)空间的向量表示,  $a_n$ 、 $b_n$  分别是  $f(x)$  在坐标轴  $\cos nx$  和  $\sin nx$  上的(投影)分量。

我们知道, 在三维空间的表示中  $\vec{i}$ 、 $\vec{j}$ 、 $\vec{k}$  是相互正交的, 即

$$(\vec{i}, \vec{j}) = (\vec{i}, \vec{k}) = (\vec{j}, \vec{k}) = 0, \quad (\vec{i}, \vec{i}) = (\vec{j}, \vec{j}) = (\vec{k}, \vec{k}) = 1$$

若把三角级数看作一种向量表示, 自然也应具有正交性。

## C.3 以 $2\pi$ 为周期函数的 Fourier 级数

三角级数的系数与其和函数的关系如下所述。

若在整个数轴上有





$$f(x) = \frac{a_0}{2} + \sum_{n=1} a_n \cos nx + b_n \sin nx$$

且等式右端的级数一致收敛，则有

$$a_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos nx dx, \quad n = 0, 1, 2, \dots$$

$$b_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \sin nx dx, \quad n = 1, 2, \dots$$

设函数  $f(x)$  在区间  $[-\pi, \pi]$  上可积，则称

$$a_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos kx dx, \quad k = 0, 1, 2, \dots$$

$$b_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \sin kx dx, \quad k = 1, 2, \dots$$

为 Euler-Fourier 公式；称由 Euler-Fourier 公式得到的  $a_n$  和  $b_n$  为函数  $f(x)$  的 Fourier 系数；并称以 Fourier 系数  $a_n$  和  $b_n$  为系数的三角级数

$$\frac{a_0}{2} + \sum_{n=1} a_n \cos nx + b_n \sin nx$$

为函数  $f(x)$  的 Fourier 级数，记为

$$f(x) = \frac{a_0}{2} + \sum_{n=1} a_n \cos nx + b_n \sin nx$$

【例 C-1】设  $f(x) = x, x \in [-\pi, \pi]$ ，求函数  $f(x)$  的 Fourier 级数。

$f(x)$  是  $[-\pi, \pi]$  上的奇函数，则有  $a_k = 0$ ，及

$$\begin{aligned} b_k &= \frac{1}{\pi} \int_{-\pi}^{\pi} x \sin kx dx \\ &= \frac{2}{\pi} \int_0^{\pi} x \sin kx dx \\ &= \frac{2}{\pi} \left( -\frac{x \cos kx}{k} \Big|_0^{\pi} + \frac{1}{k} \int_0^{\pi} \cos kx dx \right) \\ &= \frac{(-1)^{k-1} 2}{k} \end{aligned}$$

因此， $f(x) = 2 \sum_{n=1} (-1)^{n-1} \frac{\sin nx}{n}$ 。

## C.4 Fourier 变换

Fourier 级数主要表征的是周期信号的性质，但是工程应用中有大量的非周期信号，因此，引入了 Fourier 变换对非周期信号进行分析。

经典的 Fourier 变换 (FT) 定义如下：

$$F(\omega) = \int_{-\infty}^{+\infty} f(t) e^{-j\omega t} dt \quad (\text{D-4})$$



$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} F(\omega) e^{j\omega t} d\omega \quad (\text{C-5})$$

其中, 式 (C-5) 称为 Fourier 逆变换 (IFT)。

当然, 传统的 Fourier 定义中指出, 原信号  $f(t)$  还必须满足狄里赫里条件。对于大多数工程信号来说, 这个条件是容易满足的。

在式 (C-4) 中,  $F(\omega)$  实际上是对原信号  $f(t)$  做了频谱分析。对于某个固定的频率  $\omega = \omega_1$ , 变换结果如下:

$$F(\omega) \Big|_{\omega=\omega_1} = F(\omega_1) = \int_{-\infty}^{+\infty} f(t) e^{-j\omega_1 t} dt \quad (\text{C-6})$$

式 (C-6) 给出了变换  $F(\omega)$  在  $\omega = \omega_1$  处频率的大小, 可以理解为原信号中包含频率分量  $e^{j\omega_1 t}$  的大小。因此, 对非周期信号  $f(t)$ , 通过 Fourier 变换可以方便地分析各个频率分量。需要注意的是  $F(\omega)$  中关于频率  $\omega$  的连续函数, 也就意味着它表示的频率成分是丰富的, 而不仅仅限于  $k$  次谐波, 这与 Fourier 级数是不同的。

另外, Fourier 变换也可以认为是对信号进行了相干积累。根据 Fourier 变换性质, 有

$$F(\omega) \Big|_{\omega=0} = \int_{-\infty}^{+\infty} f(t) dt \quad (\text{C-7})$$

它既是原信号的零频分量, 也是原信号在零频处的相干累积。

而在某个固定的频率位置  $\omega = \omega_1$ , 乘积  $f'(t) = f(t) e^{-j\omega_1 t}$  实际上是将信号  $f(t)$  的频谱左移  $\omega_1$  位置。再通过  $\int_{-\infty}^{+\infty} \cdot dt$  进行相干累积, 结果得到了原信号  $f(t)$  在频率位置  $\omega = \omega_1$  处相干累积的大小, 即

$$\text{Coh} = F(\omega_1) = \int_{-\infty}^{+\infty} f'(t) dt = \int_{-\infty}^{+\infty} f(t) e^{-j\omega_1 t} dt \quad (\text{C-8})$$

因此, 此相干积累结果就是  $f(t)$  在  $\omega_1$  频率处的分量大小。

## C.5 Fourier 变换及 MATLAB 实现

信号  $f(t)$  的 Fourier 变换定义为

$$F(\omega) = \int_{-\infty}^{+\infty} f(t) e^{-j\omega t} dt$$

值得注意的是,  $f(t)$  的 Fourier 变换存在的充分条件是  $f(t)$  在无限区间内绝对可积, 即  $f(t)$  满足

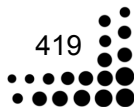
$$\int_{-\infty}^{+\infty} |f(t)| dt < \infty \quad (\text{C-9})$$

但式 (C-9) 并非  $f(t)$  存在的必要条件。当引入奇异函数概念后, 使一些不满足绝对可积的  $f(t)$  也能进行 Fourier 变换。

Fourier 逆变换定义是

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} F(\omega) e^{j\omega t} d\omega \quad (\text{C-10})$$

MATLAB 的 Symbolic Math Toolbox 提供了能直接求解 Fourier 变换及其逆变换的函数 `fourier()` 和 `ifourier()`。两者调用格式如下。





## 1. Fourier 变换

```
F=fourier(f)
F=fourier(f,v)
F=fourier(f,u,v)
```

说明:

$F=fourier(f)$  是符号函数  $f$  的 Fourier 变换, 默认返回是关于  $\omega$  的函数。如果  $f = f(\omega)$ , 则  $fourier$  函数返回关于  $t$  的函数。

$F=fourier(f,v)$  返回函数  $F$  关于符号对象  $v$  的函数, 而不是默认的  $\omega$ , 即

$$F(v) = \int_{-\infty}^{+\infty} f(t) e^{-jvt} dt$$

$F=fourier(f,u,v)$  对关于  $u$  的函数  $f$  进行变换, 返回函数  $F$  关于  $v$  的函数, 即

$$F(v) = \int_{-\infty}^{+\infty} f(u) e^{-jvt} dt$$

## 2. Fourier 逆变换

```
f=ifourier(F)
f=ifourier(F,u)
f=ifourier(F,v,u)
```

说明:

$f=ifourier(F)$  是函数  $F$  的 Fourier 逆变换。默认的独立变量为  $\omega$ , 默认返回关于  $x$  的函数。如果  $F = F(x)$ , 则  $ifourier$  函数返回关于  $t$  的函数。

$f=ifourier(F,u)$  返回函数  $f$  是  $u$  的函数, 而不是默认  $x$  的函数。

$f=ifourier(F,v,u)$  对关于  $v$  的函数  $F$  进行逆变换, 返回关于  $u$  的函数。

注意: 在调用  $fourier()$  和  $ifourier()$  之前, 要用 `syms` 命令对所有用到的变量 (如  $t$ 、 $u$ 、 $v$ 、 $w$ ) 等进行说明, 即要将这些变量说明成符号变量。对  $fourier()$  中的函数  $f$  及  $ifourier()$  中的函数  $F$  也要用符号定义符 `syms` 将  $f$  或  $F$  说明为符号表达式; 若  $f$  或  $F$  是 MATLAB 中的通用函数表达式, 则不必用 `syms` 加以说明。

## C.6 MATLAB 函数实现 Fourier 变换

【例 C-2】求  $f(t) = e^{-2|t|}$  的 Fourier 变换。

利用如下程序实现:

```
syms t
fourier(exp(-2*abs(t)))
ans =
4/(4+w^2)
```

若 Fourier 变换的结果变量希望是  $v$ , 则可运行如下程序:

```
syms t v
fourier(exp(-2*abs(t)),t,v)
```

```
ans =
4/(4+v^2)
```

【例 C-3】求  $F(\omega) = \frac{1}{1+\omega^2}$  的 Fourier 逆变换  $f(t)$ 。

利用如下程序实现：

```
syms t w
ifourier(1/(1+w^2),t)
ans =
1/2*exp(-t)*Heaviside(t)+1/2*exp(t)*Heaviside(-t)
```

其中，Heaviside(t)即为单位阶跃函数  $u(t)$ 。

【例 C-4】设  $f(t) = \frac{1}{2}e^{-2t}u(t)$ ，试画出  $f(t)$  及其幅频图。

程序如下：

```
syms t v w x;
x=1/2*exp(-2*t)*sym('Heaviside(t)');
F=fourier(x);
subplot(211);
ezplot(x);
subplot(212);
ezplot(abs(F));
```

程序运行结果如图 C-3 所示。

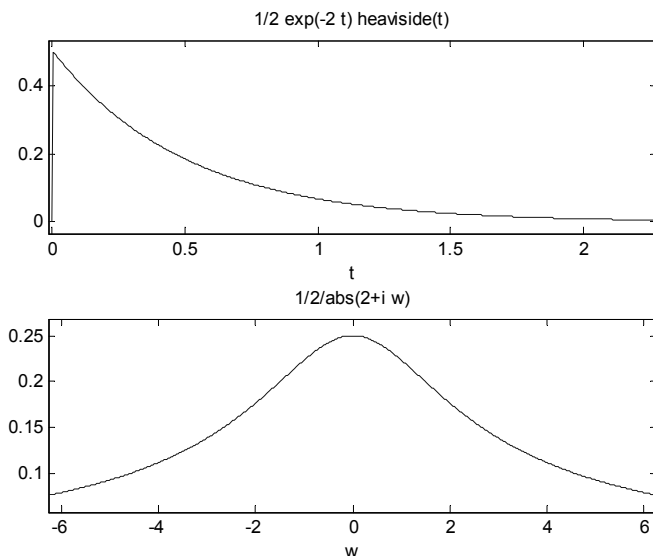


图 C-3 例 C-4 程序运行结果

本程序中的 Heaviside(t)调用了 Symbolic Math Toolbox 的 Heaviside.m 文件，内容为

```
function f= Heaviside(t)
```



$f=(t>0);$

注意：采用 `fourier()` 和 `ifourier()` 得到的返回函数，仍然是符号表达式。若需对返回函数作图，则应用 `ezplot()` 绘图命令而不能用 `plot()` 命令。如果返回函数中有诸如狄拉克函数  $\delta(\omega)$  等项，则用 `ezplot()` 也无法作图。用 `fourier()` 对某些信号求变换时，其返回函数可能会包含一些不能直接表达的式子，甚至可能会出现一些屏幕提示“未被定义的函数或变量”的项，更不用说对此返回函数作图了。这是 `fourier()` 的一个局限。另一个局限是，在很多场合，原信号  $f(t)$  尽管是连续的，但却不可能表示成符号表达式，而更多的实际测量现场获得的信号是多组离散的数值量  $f(k)$ ，此时也不可能应用 `fourier()` 对  $f(k)$  进行处理，而只能用下面介绍的数字计算方法求解。

## C.7 连续时间信号 Fourier 变换的数值计算

为了更好地体会 MATLAB 的数值计算功能，特别是强大的矩阵运算能力，这里给出连续信号 Fourier 变换的数值计算方法。方法的理论依据为

$$F(\omega) = \int_{-\infty}^{+\infty} f(t)e^{-j\omega t} dt = \lim_{\tau \rightarrow 0} \sum_{n=-\infty}^{\infty} f(n\tau)e^{-j\omega n\tau} \quad (\text{C-11})$$

对于一大类信号，当取  $\tau$  足够小时，式 (C-11) 的近似情况可以满足实际需要。若信号  $f(t)$  是时限的，或当  $|t|$  大于某个给定值时， $f(t)$  的值已经衰减得很厉害，可以近似地看成时限信号，则式 (C-11) 中的  $n$  取值就是有限的，设为  $N$ ，有

$$F(k) = \tau \sum_{n=-N}^{N-1} f(n\tau)e^{-j\omega_k n\tau}, \quad 0 \leq k \leq N-1 \quad (\text{C-12})$$

式 (C-12) 是对式 (C-11) 中的频率  $\omega$  进行取样，通常有

$$\omega_k = \frac{2\pi}{N\tau} k \quad (\text{C-13})$$

采用 MATLAB 实现式 (C-12) 时，其要点是要正确生成  $f(t)$  的  $N$  个样本  $f(n\tau)$  的向量  $f$  及向量  $e^{-j\omega_k n\tau}$ ，两向量的内积（即两矩阵相乘）结果即为式 (C-13) 的计算结果。

此外，还要注意取样周期  $\tau$  的确定。其依据是  $\tau$  需小于奈奎斯特取样周期。如果对于某个信号  $f(t)$ ，它不是严格的带限信号，则可根据实际计算的精度要求来确定一个适当的频率  $\omega_0$  为信号的带宽。

**【例 C-5】** 已知门信号  $f(t) = G_2(t) = \begin{cases} 1, & |t| < 1 \\ 0, & |t| > 1 \end{cases}$ ，求其 Fourier 变换  $F(\omega)$ 。

由信号分析可知，该信号的频谱为  $F(\omega) = 2\text{Sa}(\omega)$ ，其第一个过零点频率为  $\pi$ ，一般将此频率认为是信号  $f(t)$  的带宽。考虑到  $F(\omega)$  的形状，将精度提高到该值的 50 倍，即  $\omega_0 = 50\pi$ ，据此确定取样

$$\tau < \frac{1}{2f_0} = \frac{1}{2 \times \frac{\omega_0}{2\pi}} = 0.02$$

实现该过程的程序如下：

```

R=0.02;t=-2:R:2;
f=Heaviside(t+1)-Heaviside(t-1);
W1=2*pi*5;
N=500;k=0:N;W=k*W1/N;
F=f*exp(-j*t'*W)*R;
F=real(F);
W=[-fliplr(W),W(2:501)];
F=[fliplr(F),F(2:501)];
subplot(2,1,1);plot(t,f);
xlabel('t');ylabel('f(t)');
title('f(t)=u(t+1)-u(t-1)');
subplot(2,1,2);plot(W,F);
xlabel('w');ylabel('F(w)');
title('f(t)的付氏变换 F(w)');

```

程序运行结果如图 C-4 所示。

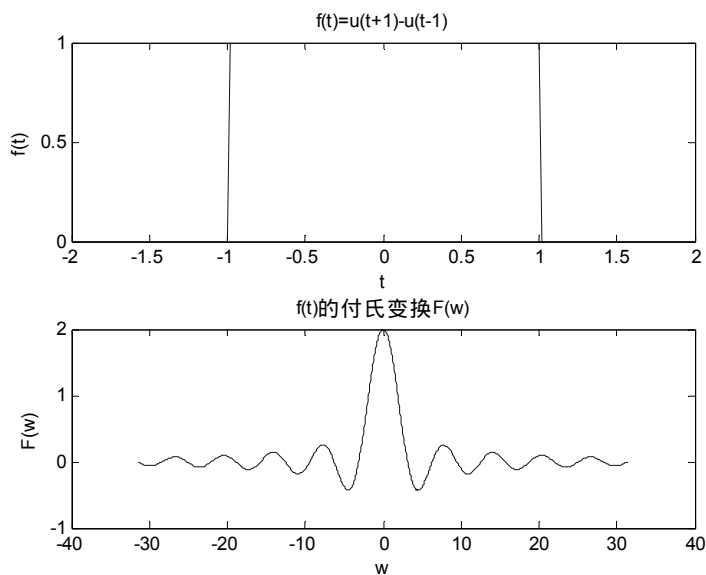


图 C-4 矩形脉冲信号的 Fourier 变换

## C.8 信号的 Fourier 分解与合成 MATLAB 实现

【例 C-6】将振幅为 1 的 1Hz 正弦波和振幅为 0.5 的 5Hz 正弦波相叠加进行分析，研究能否从中分析出含有这两种频率的信号。

程序如下：

```

clear all          %将工作空间中的所有变量清除
N=256;dt=0.02;    %数据的个数和采样周期

```





```

n=0:N-1;t=n*dt;           %序号序列和时间序列
x=sin(2*pi*t)+0.5*sin(2*pi*5*t); %信号加得到的合成信号
m=floor(N/2)+1;           %分解 a、b 的最大序号值，为分解的 N/2 个参数再加参数 a0
%floor 函数为向下取整
a=zeros(1,m);b=zeros(1,m); %产生 a、b 两个为零的序列
for k=0:m-1
    for ii=0:N-1
        a(k+1)=a(k+1)+2/N*x(ii+1)*cos(2*pi*k*ii/N);
        b(k+1)=b(k+1)+2/N*x(ii+1)*sin(2*pi*k*ii/N);
    %MATLAB 中的数组序号只能从 1 开始
    end
    c(k+1)=sqrt(a(k+1).^2+b(k+1).^2);
end
subplot(2,1,1),plot(t,x);title('原始信号'),xlabel('时间/s')           %绘出时间域信号
subplot(2,1,2),plot((0:m-1)/(N*dt),c)                                %绘出频率域信号
title('Fourier 变换'),xlabel('频率/Hz'),ylabel('振幅')

```

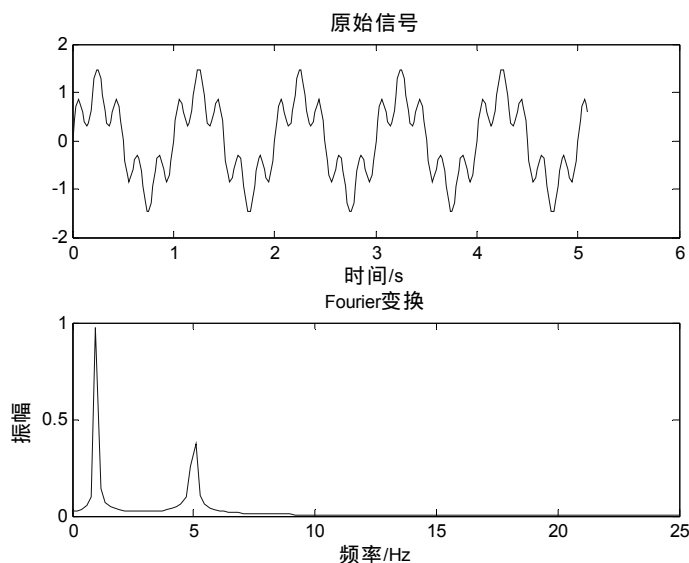


图 C-5 分析含有这两种频率的信号

程序运行结果如图 C-6 所示,可以清楚地看出,从初始信号中明确地识别出了 1Hz 和 5Hz 的波。这里 1Hz 和 5Hz 的振幅与原来信号振幅并不完全一致,这是数据采样点较少造成的。自己做实验可以看出,数据采样点越多,Fourier 分析得到的结果与原始振幅越接近。

如果将  $m$  修改为  $N$ ,其频谱只是增加了镜像对称的部分;如果将  $m$  改为  $2*N$ ,则频谱增加了 1 个周期,见图 C-6;改为  $3*N$ ,频谱就增加了 3 个周期,可以明确地看出频谱的周期性。由此可以看出,离散有限信号的频谱为周期谱。

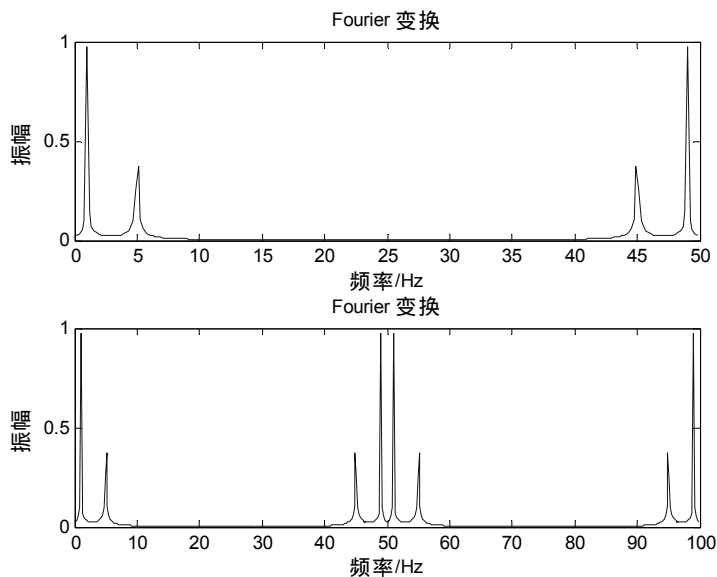


图 C-6 取大于奈奎斯特频率时振幅谱图

本例表明， $k$  值取不小于  $N/2$  时，可以完全分辨出小于采样频率一半（奈奎斯特频率）的频率；反过来，根据分解成的  $a_k$ 、 $b_k$  值能否准确地恢复原来的信号呢？下面采用  $a_k$ 、 $b_k$  的值求得本例合成的信号，在例 C-7 程序运行后，运行下面的程序：

```
if(mod(N,2)~=1)a(m)=a(m)/2; end      %此时 b(m)为零，a(m)减半
for ii=0:N-1
    xx(ii+1)=a(1)/2;
    for k=1:m-1
        xx(ii+1)=xx(ii+1)+a(k+1)*cos(2*pi*k*ii/N)+b(k+1)*sin(2*pi*k*ii/N);
    %
    end
end
subplot(2,1,1)
plot((0:N-1)*dt,x)                %绘制原信号
title('原始信号')
subplot(2,1,2)
plot((0:N-1)*dt,xx)                %绘制合成信号
title('合成信号'),xlabel('时间/s')
```

程序运行结果如图 C-7 所示，可以看到，运用 Fourier 分解得到的系数合成后的振动和原振动完全一致。验证了上述理论的正解性。

我们从另一个角度来考虑问题，如果  $k$  取不到  $\text{floor}(N/2)+1$ ，所合成的波形与原波形有何区别呢？下面以方波为例来说明这个问题。

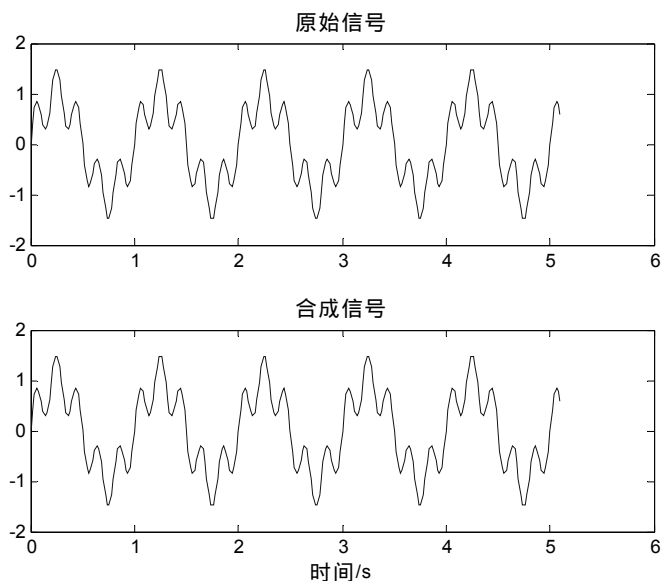


图 C-7 Fourier 分解系数合成后的振动

【例 C-7】用振幅为 0.8 的方波进行 Fourier 分析，并用分析得到的系数求解当  $k$  取不同值时的合成信号图。

程序如下：

```
clear all                %清除内存所有变量
close all                %关闭所有打开的图形窗口
N=200;dt=4/N;           %数据点数和采样间隔
for n=1:N                %得到方波信号
    if (n*dt>=2)
        x(n)=0.8;
    else
        x(n)=-0.8;
    end
end
figure(1)                %打开第一个图形窗口，绘制原始信号及其 Fourier 分析系数
subplot(2,1,1),plot((1:N)*dt,x),hold on;
plot((1:N)*dt,zeros(1,N),'k'),xlabel('时间/s')
title('原始信号')
a=zeros(1,N);b=zeros(1,N);
nn=floor(N/2)+1;
for k=0:nn-1
    a(k+1)=0;
    b(k+1)=0;
    for ii=0:N-1
        a(k+1)=a(k+1)+2/N*x(ii+1)*cos(2*pi*k*ii/N);    %求解 Fourier 系数
        b(k+1)=b(k+1)+2/N*x(ii+1)*sin(2*pi*k*ii/N);
```

```

end
c(k+1)=sqrt(a(k+1).^2+b(k+1).^2);
end
subplot(2,1,2),plot((0:nn-1)/(N*dt),c);title('Fourier 变换')
xlabel('频率/Hz'),ylabel('振幅')           %绘制振幅谱
m=input('输入谐波最大阶数?');           %输入最大 k 值
if(m>(floor(N/2)+1))                     %如果最大 k 值大于奈奎斯特频率对应的点数，则显示出错信息
    error('谐波最大阶数必须小于奈奎斯特频率对应的阶数.')
end
if(mod(N,2)~=1)a(nn)=a(nn)/2; end        %此时 b(nn)为零，a(nn)减半
for ii=0:N-1    %合成信号
    xx(ii+1)=a(1)/2;
    for k=1:m
        xx(ii+1)=xx(ii+1)+a(k+1)*cos(2*pi*k*ii/N)+b(k+1)*sin(2*pi*k*ii/N);
    end
end
figure(2)                                     %打开第二个图形窗口，绘制合成信号图
plot((1:N)*dt,xx,(0:N-1)*dt,x)              %绘制合成信号和方波图形便于比较
hold on
plot((1:N)*dt,zeros(1,N),'k'),xlabel('时间/s') %绘出横轴
title('合成信号')

```

程序运行后得到的第一个图如图 C-8 所示。该图表明方波信号进行 Fourier 分解后，得到的系数几乎分布在整个频率轴上，即说明方波可以分解为含有多种频率的振动。从图 C-8 中还可以发现，低频成分的振幅较大，随着频率的增高，振幅逐渐衰减。

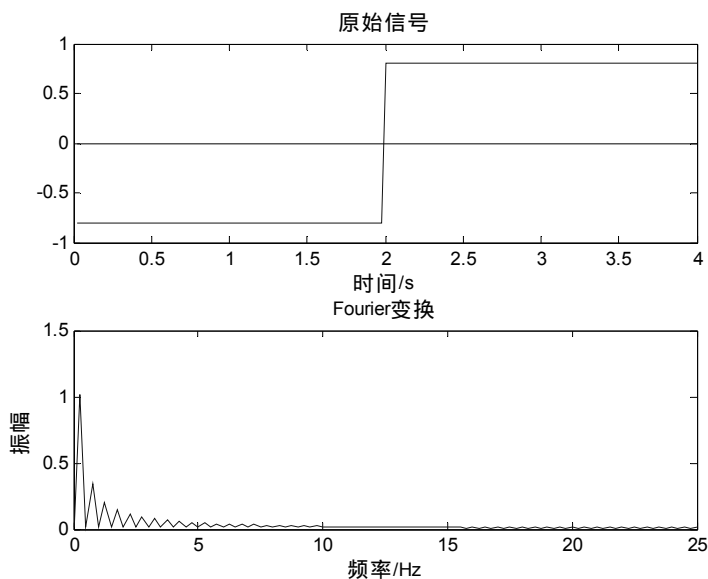


图 C-8 方波 Fourier 分析



当  $m=1, 3, 5, 7$  时, 得到的合成信号组合在一起, 如图 C-9 所示, 可以看出,  $m$  越大, 即  $k$  取得越多, 与原始的方波信号越接近。当  $m=1$  时, 只将方波分解为一个正弦振动; 当  $m=3$  时, 在原来  $m=1$  振动的基础上又叠加了一次谐波, 使得该振动波形与方波更加接近; 当  $m=5$  时, 得到的振动波形与方波更加接近, 依次类推。可以想象, 当  $m=\text{floor}(N/2)+1$  时, 合成波形与原波形完全一致。本例中的方波信号为奇函数, 只有  $b_k$  在起作用。

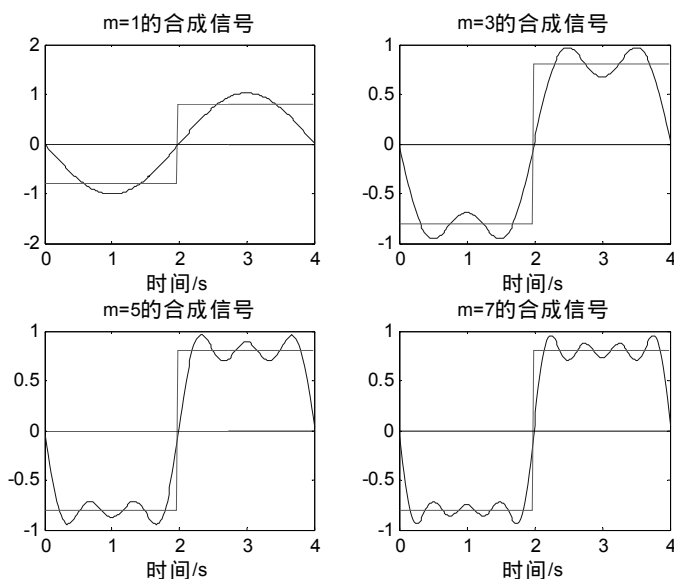


图 C-9 运用分解得到 Fourier 系数合成原振动图像

通过前面的 Fourier 级数分析, 我们知道采用 Fourier 级数分解技术可以得到信号含有哪种频率成分、振幅为多少, 并且运用分解得到 Fourier 系数可以合成为原振动图像, 这是 Fourier 分析的基础。

## 附录 D 快速 Fourier 变换及其应用

### D.1 快速 Fourier 变换及其 MATLAB 应用

有限长序列可以通过离散 Fourier 变换 (DFT) 将其频域也离散化成有限长序列。但其计算量太大, 很难实时地处理问题, 因此引出了快速 Fourier 变换 (FFT)。1965 年, Cooley 和 Tukey 提出了 DFT 的快速算法, 将 DFT 的运算量减少了几数量级。从此, 对 FFT 算法的研究便不断深入, 数字信号处理这门新兴学科也随 FFT 的出现和发展而迅速发展, 根据对序列分解与选取方法的不同而产生了 FFT 的多种算法, 基本算法是基 2 DIT 和基 2 DIF。FFT 在离散 Fourier 反变换、线性卷积和线性相关等方面也有重要应用。

FFT 是计算 DFT 的快速算法。

DFT 的定义式为

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn} R_N(k)$$

在所有复指数值  $W_N^{kn}$  的值全部已算好的情况下, 要计算一个  $X(k)$  需要  $N$  次复数乘法和  $N-1$  次复数加法。算出全部  $NX(k)$  点共需  $N^2$  次复数乘法和  $N(N-1)$  次复数加法, 即计算量是与  $N^2$  成正比的。

FFT 的基本思想: 将大点数的 DFT 分解为若干个小点数 DFT 的组合, 从而减少运算量。

$W_N$  因子具有以下两个特性, 可使 DFT 运算尽量分解为小点数的 DFT 运算。

(1) 周期性:  $W_N^{(k+N)n} = W_N^{kn} = W_N^{(n+N)k}$ ;

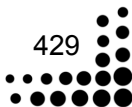
(2) 对称性:  $W_N^{(k+N/2)} = -W_N^k$ 。

利用这两个性质, 可以使 DFT 运算中有些项合并, 以减少乘法次数。例如, 求  $N=4$  时,  $X(2)$  的值:

$$\begin{aligned} X(2) &= \sum_{n=0}^3 x(n) W_4^{2n} = x(0) W_4^0 + x(1) W_4^2 + x(2) W_4^4 + x(3) W_4^6 \\ &= [x(0) + x(2)] W_4^0 + [x(1) + x(3)] W_4^2 \quad (\text{周期性}) \\ &= [x(0) + x(2)] - [x(1) + x(3)] W_4^0 \quad (\text{对称性}) \end{aligned}$$

通过合并, 可使乘法次数由 4 次减少到 1 次, 运算量减少。

FFT 的算法形式有很多种, 但基本上可以分为两大类: 按时间抽取 (DIT) 和按频率抽取 (DIF)。





## 1. FFT 变换的用法

在 MATLAB 信号处理工具箱中, 函数 FFT 和 IFFT 分别实现快速 Fourier 变换及其逆变换。快速 Fourier 变换函数调用格式为:

$$y = \text{FFT}(x)$$

其中,  $x$  是序列;  $y$  是序列的快速 Fourier 变换。 $x$  可以为一向量或矩阵, 若  $x$  向量, 则  $y$  是  $x$  的 FFT, 并且与  $x$  具有相同的长度; 若  $x$  为一矩阵, 则  $y$  对矩阵的每列向量进行 FFT。

如果  $x$  的长为 2 的整数次幂, 则函数 FFT 执行高速-2FFT 算法, 否则执行一种混合基的离散 Fourier 算法, 计算速度较慢。这就是说, 只有当  $x$  的长度为 2 的整数次幂时才能最大限度地提高程序运算速度。

函数 FFT 的另一种调用形式为:

$$y = \text{FFT}(x, N)$$

其中,  $x$ 、 $y$  的意义同前;  $N$  为正整数。此时, 函数执行  $N$  点的 FFT。若  $x$  为向量且长度小于  $N$ , 则函数将  $x$  补零至长度  $N$ ; 若向量  $x$  的长度大于  $N$ , 则函数截断  $x$  使之长度为  $N$ 。

对应于快速 Fourier 变换函数 FFT, MATLAB 信号处理工具箱中提供的快速 Fourier 逆变换函数为:

$$y = \text{IFFT}(X) \text{ 和 } y = \text{IFFT}(X, N)$$

这里,  $X$  为需要进行逆变换的序号信号, 一般情况下为复数;  $y$  为快速 Fourier 逆变换的输出, 通常包含实部和虚部两部分。 $N$  的意义与 FFT 函数的一样。

用 MATLAB 进行频谱分析时注意以下几点。

(1) 函数 FFT 返回值的数据结构具有对称性。

【例 D-1】分析 (4 3 2 6 7 8 9 0) 的结构对称性

程序如下:

```
N=8;
n=0:N-1;
xn=[4 3 2 6 7 8 9 0];
Xk=fft(xn)
```

程序运行结果如下:

```
Xk =
Columns 1 through 5
39.0000    -10.7782 + 6.2929i     0 - 5.0000i    4.7782 - 7.7071i     5.0000
Columns 6 through 8
4.7782 + 7.7071i     0 + 5.0000i    -10.7782 - 6.2929i
```

(2) 做 FFT 分析时, 幅值大小与 FFT 选择的点数有关, 但不影响分析结果, 在快速 Fourier 逆变换时已经做了处理。要得到真实的振幅的大小, 只要将得到的变换结果乘以 2 除以  $N$  即可。

## 2. 快速 Fourier 变换应用举例

【例 D-2】一个信号由 15Hz、幅值为 0.5 的正弦信号和 40Hz、幅值为 2 的正弦信号组成。



数据采样频率  $F_s=100\text{Hz}$ ，试分别绘制  $N=128$  点 FFT 幅频图和  $N=1024$  点幅频图。

程序如下：

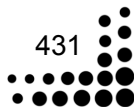
```

clf;
fs=100;N=128;                                %采样频率和数据点数
n=0:N-1;t=n/fs;                                %时间序列
x=0.5*sin(2*pi*15*t)+2*sin(2*pi*40*t);        %信号
y=fft(x,N);                                    %对信号进行快速 Fourier 变换
mag=abs(y);                                    %求得 Fourier 变换后的振幅
f=n*fs/N;                                       %频率序列
subplot(2,2,1),plot(f,mag);                    %绘出随频率变化的振幅
xlabel('频率/Hz');
ylabel('振幅');title('N=128');grid on;
subplot(2,2,2),plot(f(1:N/2),mag(1:N/2));      %绘出奈奎斯特频率之前随频率变化的振幅
xlabel('频率/Hz');
ylabel('振幅');title('N=128');grid on;
%对信号采样数据为 1024 点的处理
fs=100;N=1024;n=0:N-1;t=n/fs;
x=0.5*sin(2*pi*15*t)+2*sin(2*pi*40*t);        %信号
y=fft(x,N);                                    %对信号进行快速 Fourier 变换
mag=abs(y);                                    %求取 Fourier 变换的振幅
f=n*fs/N;                                       %频率序列
subplot(2,2,3),plot(f,mag);                    %绘出随频率变化的振幅
xlabel('频率/Hz');
ylabel('振幅');title('N=1024');grid on;
subplot(2,2,4)
plot(f(1:N/2),mag(1:N/2));                    %绘出奈奎斯特频率之前随频率变化的振幅
xlabel('频率/Hz');
ylabel('振幅');title('N=1024');grid on;

```

程序运行结果如图 D-1 所示。本例中，由于  $F_s=100\text{Hz}$ ，奈奎斯特频率为  $F_s/2=50\text{Hz}$ 。整个频谱图是以频率为对称轴的，并且可以明显识别出信号中的含有两种频率成分：15Hz 和 40Hz。由此可以知道 Fourier 变换的数据对称性。因此，利用 FFT 对信号做谱分析，只需要考查 0 到奈奎斯特频率范围的幅频特性。如果没有给出采样频率或采样周期，则分析通常对归一化频率 0 ~ 1 进行。另外，振幅的大小与所用采样点有关，采用 128 点和 1024 点的相同频率的振幅有不同的表现值，但在同一幅图中，40Hz 的振动与 15Hz 振动振幅之比均为 1 : 4，与真实振幅 0.5 : 2 是一致的。为了与真实振幅对应，需要将变换后的结果乘以 2 除以 N，后面的分析也采用这种形式。

【例 D-3】已知信号  $x(t)=0.5\sin(2\pi f_1 t)+2\sin(2\pi f_2 t)$ ，其中， $f_1=15\text{Hz}$ ， $f_2=40\text{Hz}$ ，采样频率为 100Hz，在下列情况下绘制其幅频图：数据个数  $N=32$ ，FFT 所用的采样点数  $N_{\text{FFT}}=32$ ； $N=32$ ； $N_{\text{FFT}}=128$ ； $N=136$ ； $N_{\text{FFT}}=128$ ； $N=136$ ， $N_{\text{FFT}}=512$ 。分析所用数据长度不同时对 Fourier 变换结果的影响。





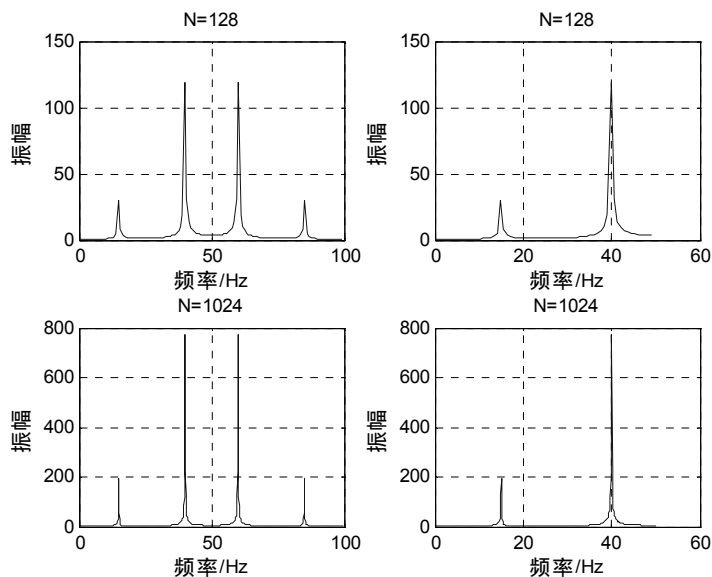


图 D-1 快速 Fourier 变换应用举例

程序如下：

```

clf;fs=100; %采样频率
Ndata=32; %数据长度
N=32; %FFT 的数据长度
n=0:Ndata-1;t=n/fs; %数据对应的时间序列
x=0.5*sin(2*pi*15*t)+2*sin(2*pi*40*t); %时间域信号
y=fft(x,N); %信号的 Fourier 变换
mag=abs(y); %求取振幅
f=(0:N-1)*fs/N; %真实频率
subplot(2,2,1),plot(f(1:N/2),mag(1:N/2)*2/N); %绘出奈奎斯特频率之前的振幅
xlabel('频率/Hz');ylabel('振幅');
title('Ndata=32 Nfft=32');grid on;

Ndata=32; %数据个数
N=128; %FFT 采用的数据长度
n=0:Ndata-1;t=n/fs; %时间序列
x=0.5*sin(2*pi*15*t)+2*sin(2*pi*40*t);
y=fft(x,N);
mag=abs(y);
f=(0:N-1)*fs/N; %真实频率
subplot(2,2,2),plot(f(1:N/2),mag(1:N/2)*2/N); %绘出奈奎斯特频率之前的振幅
xlabel('频率/Hz');ylabel('振幅');
title('Ndata=32 Nfft=128');grid on;

Ndata=136; %数据个数
N=128; %FFT 采用的数据个数
n=0:Ndata-1;t=n/fs; %时间序列
x=0.5*sin(2*pi*15*t)+2*sin(2*pi*40*t);

```



```

y=fft(x,N);
mag=abs(y);
f=(0:N-1)*fs/N;                                     %真实频率
subplot(2,2,3),plot(f(1:N/2),mag(1:N/2)*2/N);        %绘出奈奎斯特频率之前的振幅
xlabel('频率/Hz');ylabel('振幅');
title('Ndata=136 Nfft=128');grid on;

Ndata=136;                                           %数据个数
N=512;                                              %FFT 所用的数据个数
n=0:Ndata-1;t=n/fs;                                %时间序列
x=0.5*sin(2*pi*15*t)+2*sin(2*pi*40*t);
y=fft(x,N);
mag=abs(y);
f=(0:N-1)*fs/N;                                     %真实频率
subplot(2,2,4),plot(f(1:N/2),mag(1:N/2)*2/N);        %绘出奈奎斯特频率之前的振幅
xlabel('频率/Hz');ylabel('振幅');
title('Ndata=136 Nfft=512');grid on;

```

程序运行结果如图 D-2 所示。当数据个数和 FFT 采用的数据个数均为 32 时, 频率的分辨率较低, 但没有由于添加零而导致的其他频率成分。这里由于将振幅谱乘以  $2/N$ , 因此就得到了绝对大小的振幅。当数据个数为 32、FFT 采用 128 时, FFT 程序将数据后补零, 补足 128 个数据。由于在时间域内信号加零, 致使振幅谱中出现很多其他成分, 这是加零造成的。其振幅由于加了多个零而明显减小。当数据个数增加到 136 时, FFT 所用数据个数为 128, 则 FFT 程序将数据截断为 128 个, 这时分辨率较高; 当数据个数为 136 时, FFT 所用数据为 512, 也需要在原始数据后补加  $512-136$  个零。虽然如此, 但由于含有信号的数据个数足够多, 其 Fourier 振幅谱也基本不受影响。在对信号做频谱分析时, 数据样本应有足够的长度, 一般应取 FFT 程序中所用数据点数与原含有信号数据点数相同, 这样的频谱图具有较高的质量, 可减少因补零或截断而产生的影响。

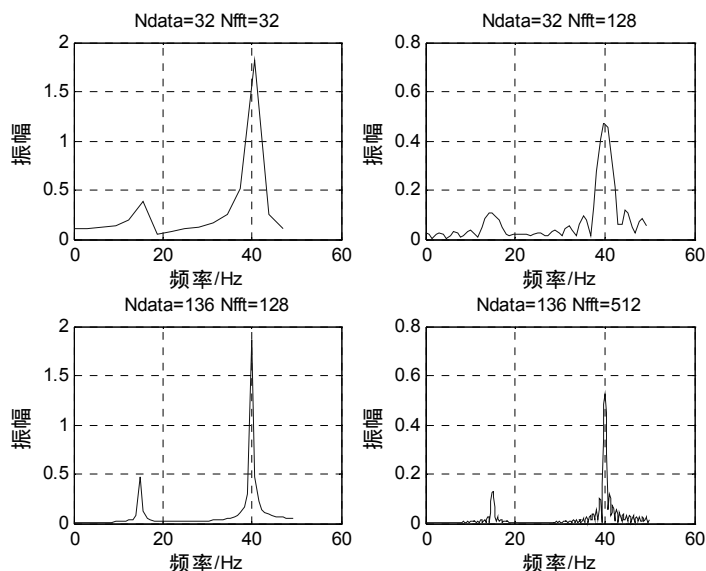
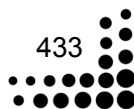


图 D-2 数据长度不同时对 Fourier 变换结果的影响





【例 D-4】已知带有测量噪声信号  $x(t)=\sin(2\pi f_1 t)+2\sin(2\pi f_2 t)+2w(t)$ ，其中， $f_1=50\text{Hz}$ ， $f_2=120\text{Hz}$ ， $w(t)$ 是均值为零的随机信号。采样频率为  $1000\text{Hz}$ ，数据点数  $N=1024$ ，试绘制信号的频谱图和无噪声信号频谱图。

程序如下：

```
clf;
fs=1000;N=1024; %采样频率和数据点数
n=0:N-1;t=n/fs; %时间序列
f1=50;f2=120; %信号的频率成分
x=sin(2*pi*f1*t)+2*sin(2*pi*f2*t); %无噪声信号
x=x+2*randn(1,length(t)); %含有噪声的信号
y=fft(x,N); %对含有噪声信号进行 Fourier 变换
mag=abs(y); %得到振幅值
f=n*fs/N; %真实频率序列
subplot(2,1,1),plot(f(1:N/2),mag(1:N/2)*2/N); %绘出奈奎斯特频率之前的振幅
xlabel('频率/Hz');
ylabel('振幅');title('N=1024');grid on;
x=sin(2*pi*f1*t)+2*sin(2*pi*f2*t); %无噪声信号
y=fft(x,N); %对无噪声信号进行快速 Fourier 变换
mag=abs(y); %求取振幅
f=n*fs/N; %真实频率序列
subplot(2,1,2),plot(f(1:N/2),mag(1:N/2)*2/N); %绘出奈奎斯特奈奎斯特频率之前的振幅
xlabel('频率/Hz');
ylabel('振幅');title('N=1024 ( 无噪声 )');grid on;
```

程序运行结果如图 D-3 所示，可以看到，无论是否加有噪声的信号，均能通过 Fourier

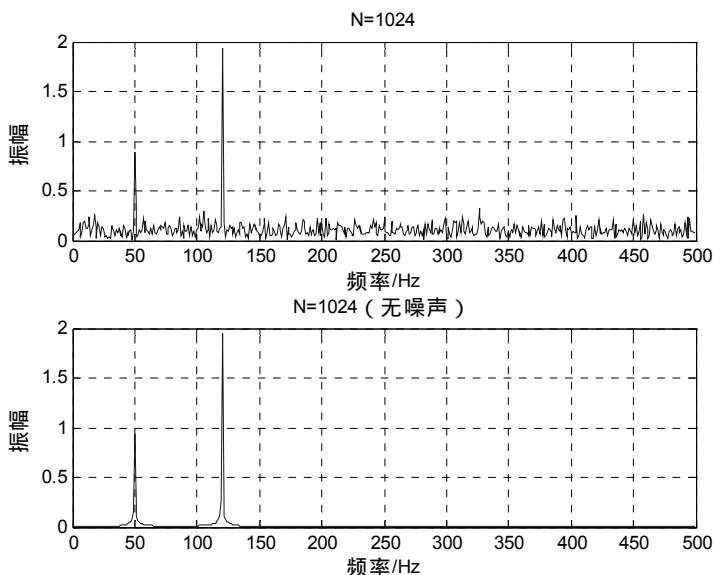


图 D-3 含有噪声和不含噪声的 Fourier 振幅谱



分析识别出其中的 50Hz 和 120Hz 的信号。本例程序中,函数 `randn(1,N)` 可以产生 1 行  $N$  列在零点附近分布的随机信号。

【例 D-5】对信号  $x(t)=\sin(2\pi\cdot 40t)+\sin(2\pi\cdot 15t)$  进行 FFT,对其结果进行逆 FFT,将结果与原信号进行比较。采样频率为 100Hz,采样点数  $N=128$ 。

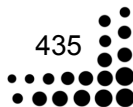
程序如下:

```
clf;
fs=100;N=128;           %采样频率和数据个数
n=0:N-1;t=n/fs;         %时间序列
x=sin(2*pi*40*t)+sin(2*pi*15*t); %时间域信号
subplot(2,2,1),plot(t,x);xlabel('时间/s');
ylabel('x');title('原始信号');
grid on;

y=fft(x,N);              %Fourier 变换
mag=abs(y);              %得到振幅谱
f=n*fs/N;                %频率序列
subplot(2,2,2),
plot(f(1:N/2),mag(1:N/2)*2/N); %绘制奈奎斯特频率前的振幅
xlabel('频率/Hz');ylabel('振幅');
title('原始信号的快速 Fourier 变换');grid on;
xifft=ifft(y);           %进行 Fourier 逆变换
realx=real(xifft);        %求取 Fourier 逆变换的实部
ti=[0:length(xifft)-1]/fs; %Fourier 逆变换的时间序列
subplot(2,2,3),plot(ti,realx);
xlabel('时间/s');ylabel('x');
title('运用 Fourier 逆变换得到的信号');grid on;

yif=fft(xifft,N);         %将 Fourier 逆变换得到的时间域信号进行 Fourier 变换
mag=abs(yif);
f=[0:length(y)-1]*fs/length(y); %频率序列
subplot(2,2,4),plot(f(1:N/2),mag(1:N/2)*2/N); %绘制奈奎斯特频率前的振幅
xlabel('频率/Hz');ylabel('振幅');
title('运用 IFFT 得到信号的快速 Fourier 变换');grid on;
```

程序运行结果如图 D-4 所示,可以看到,对 Fourier 变换后的频谱进行逆变换,得到的时间域信号与原始时间域信号完全一致。程序中对其进行取实部运算,如果大家分析一下其虚部,可以发现基本为零,不为零是由于数据截断造成的。如果数据无限长,则可得为零的结果。得到的时间域信号再次进行 Fourier 变换后与原始信号的 Fourier 变换结果也一致。



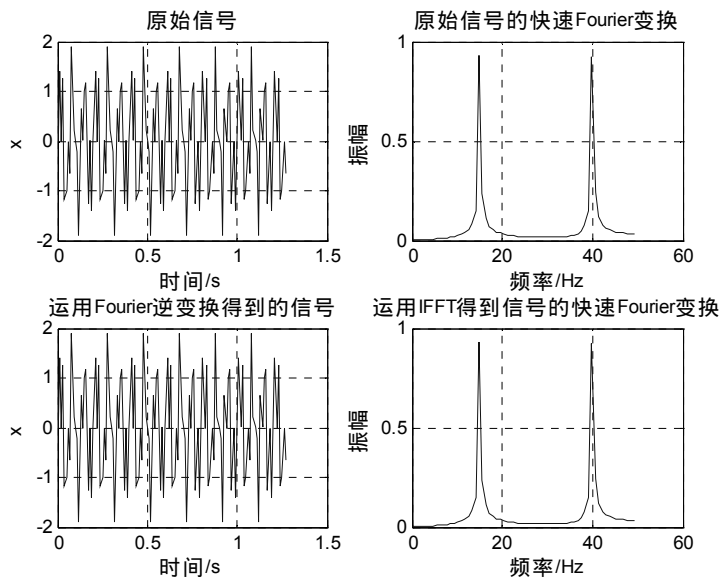


图 D-4 逆变换结果与原信号比较

【例 D-6】考虑信号  $x(n)=\cos(2\pi\cdot 0.24n)+\cos(2\pi\cdot 0.26n)$ ， $0 \leq n \leq 100$ ，取  $x(n)$  为输入信号时，求  $x(n)$  的快速 Fourier 变换；将  $x(n)$  以补零方式使  $x(n)$  加长到  $0 \leq n \leq 100$ ，求 Fourier 变换；取  $x(n)$ ， $0 \leq n \leq 100$ ，求 Fourier 变换。分析高密度频谱和高分辨频谱之间的差异。

程序如下：

```

N1=10;dt=1; %采样点数和采样间隔
n1=0:N1-1;t1=n1*dt; %时间序列
x=cos(2*pi*0.24*t1)+cos(2*pi*.26*t1); %原始信号
subplot(2,3,1);stem(t1,x); hold on;
plot([0 10],[0 0]); %绘制横轴
title('信号 x(n), n=0-9');xlabel('时间/s')
Y1=fft(x);magY1=abs(Y1);
f1=n1/(N1*dt);
subplot(2,3,4);plot(f1(1:N1/2),magY1(1:N1/2)*2/N1); %奈奎斯特频率前的振幅
xlim([0 0.5]);
title('10 个数据点信号的 FFT');
xlabel('频率/Hz')

N3=100;dt=1; %采用 10 个数据点和 90 个零点的情况
n3=0:N3-1;t3=n3*dt;
x=cos(2*.24*pi*t1)+cos(2*0.26*pi*t1);
y3=[x(1:10) zeros(1,90)];
subplot(2,3,2);stem(t3,y3);
title('含有 90 个零的 100 个数据点信号');xlabel('时间/s')
Y3=fft(y3);magY3=abs(Y3);
f=n3/(N3*dt);

```



```
subplot(2,3,5);plot(f(1:N3/2),magY3(1:N3/2)*2/N3);
xlim([0 0.5]);
title('含有 90 个零的 100 个数据点的 FFT');xlabel('频率/Hz')
n=0:N3;t=n*dt;                                     %采用 100 个数据点的情况
x=cos(2*0.24*pi*t)+cos(2*0.26*pi*t);
subplot(2,3,3);stem(t,x);
title('信号 x(n), n=0-99');xlabel('时间/s')
X=fft(x);magX=abs(X);
f=n/(N3*dt);
subplot(2,3,6);plot(f(1:N3/2),magX(1:N3/2)*2/N3);
xlim([0 0.5]);
title('100 个数据点信号的 FFT');xlabel('频率/Hz')
```

程序运行结果如图 D-5 所示。最左边的两个图为  $0 \leq n \leq 100$  时序列  $x(n)$  的 Fourier 变换，从图中几乎无法看出有关信号频谱的详细信息，这是数据点过少的缘故。中间的两个图是将  $x(n)$  补 90 个零时信号的频谱。显然，这时振幅谱的数据相当密，这是补加 90 个零的缘故，这种谱称为高密度频谱图，但从图 E-5 中很难看出信号的频谱成分。最右边两个图为加长取样长度的结果，这时可清楚地看出信号的频谱成分，一个频率为 0.24Hz，另一个频率为 0.26Hz，这称为高分辨频谱。由此可见，如果采样数据过少，则运用 Fourier 变换不能分辨出其中的频率成分；添加零后可增加频谱中的数据个数，谱的密度增高了，但仍不能分辨其中的频率成分，即谱的分辨率没有提高；只有数据点数足够多时才能分辨其中的频率成分。

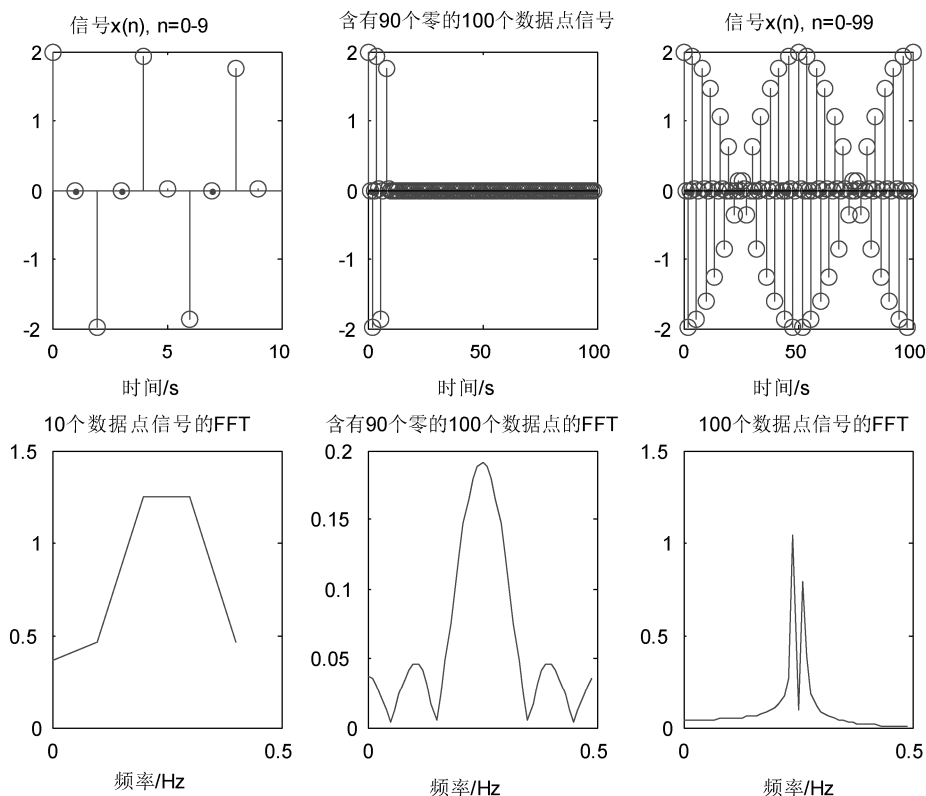
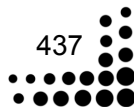


图 D-5 不同数据点数得到高密度谱和高分辨率谱的例子





## D.2 运用 FFT 进行简单滤波

由前面分析可知,根据快速 Fourier 变换我们可以知道信号序列中有哪些频率成分,各个频率成分的振幅是多大。根据快速 Fourier 变换,我们可以把频率域的信号转化回时间域,从而得到与原信号长度相同的时间序列。那么,我们能否通过将频率域中的某些频率成分振幅置零后运用 Fourier 逆变换到时间域而达到滤波的效果呢?回答是肯定的。这时一个自然提出的问题是,若将某些频率的振幅置为零,其相位信息不变,会不会有问题?可以想到,若该频率信号的振幅为零,其相位根本不起作用。但要注意,由于 FFT 得到的频率域一般只考虑奈奎斯特频率之前的频率,但当我们采用 FFT 滤波时,必须考虑奈奎斯特频率之后的振幅及相位。下面举例说明这种技术。

【例 D-7】运用 FFT 对信号  $x=0.5\sin(2\pi\cdot 3n\cdot dt)+\cos(2\pi\cdot 10n\cdot dt)$ , 数据点数为 512, 进行滤波, 将频率为 8~15Hz 的波滤去。采样周期  $dt=0.02$ 。绘出滤波前和滤波后的振幅谱以及滤波后的时间域信号。

程序如下:

```
dt=0.02;N=512;
n=0:N-1; t=n*dt; f=n/(N*dt);           %时间序列及频率序列
f1=3; f2=10;                             %信号的频率成分
x=.5*sin(2*pi*f1*t)+cos(2*pi*f2*t);
subplot(2,2,1), plot(t,x);               %绘制原来的信号
title('原始信号的时间域'); xlabel('时间/s');
y=fft(x);                                %对原信号作 FFT 变换
subplot(2,2,2), plot(f, abs(y)*2/N)      %绘制原信号的振幅谱
xlabel('频率/Hz'), ylabel('振幅')
xlim([0 50]); title('原始振幅谱')
f1=8; f2=15;                             %要滤去频率的上限和下限
yy=zeros(1,length(y));                  %设置与 y 相同元素的数组
for m=0:N-1                              %将频率落在该频率范围及大于奈奎斯特频率的波滤去
    if(m/(N*dt)>f1&m/(N*dt)<f2)...       %小于奈奎斯特频率的滤波范围
        |(m/(N*dt)>(1/dt-f2)&m/(N*dt)<(1/dt-f1))    %大于奈奎斯特频率的滤波范围
        %1/dt 为一个频率周期
        yy(m+1)=0.;                     %置在此频率范围内的振动振幅为零
    else
        yy(m+1)=y(m+1);                 %其余频率范围的振动振幅不变
    end
end
subplot(2,2,4), plot(f,abs(yy)*2/N)      %绘制滤波后的振幅谱
xlim([0 50]); xlabel('频率/Hz'); ylabel('振幅')
```



```
gtext=sprintf('自 8.0 ~ 15.0 Hz 的频率被滤除',f1,f2); %将滤波范围显示作为标题
title(gtext)
subplot(2,2,3),plot(t,real(ifft(yy))) %绘制滤波后的数据运用 IFFT 变换回时间域并绘图
title('通过 IFFT 回到时间域');
xlabel('时间/s');
```

程序运行结果如图 D-6 所示。由图可见，无论在时间域或频率域均滤除了 8 ~ 15Hz 的频率成分。大家可以选择不同的滤波范围进行试验或设计其他的信号进行试验。时间域显示其滤波效果还是相当好的。这是最彻底、最干净的滤波。这种滤波的缺点是由于使用 Fourier 变换，相比于后面所讲的滤波技术，运算相对较慢。

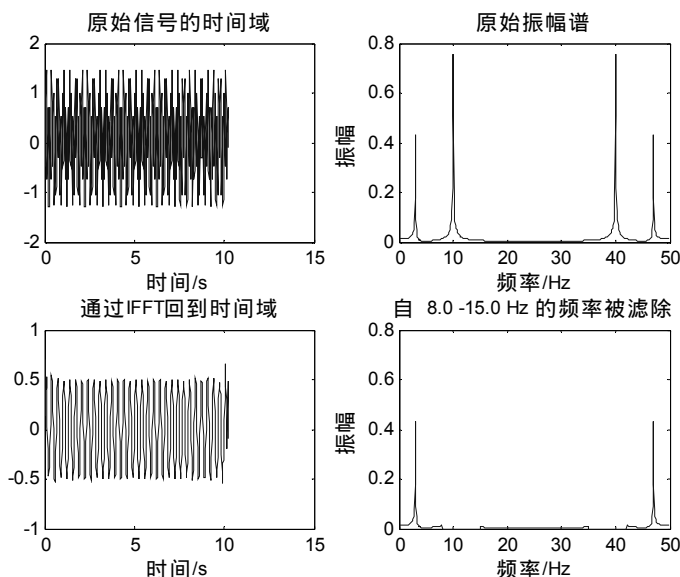
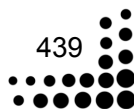


图 D-6 运用 Fourier 变换滤除 8 ~ 15Hz 频率成分

将该程序用于实例数据分析中，我们采用了某地电台电导率的 1994—1997 年日测值资料。我们知道，地电记录除了能记录地震等地球物理特征外还能记录到由于气候的年周期变化的信息，这些信息与地球物理过程没有关系，会干扰对地球物理过程的分析。因此，如果滤除气候年周期变化的信息，就可以更清晰地分析其他地球物理过程的信息。下面的程序把周期大于 200 天的信息滤去（周期为 200 天对应的频率为其倒数，即  $1/200$ ）：

```
clf
load w9497d; %加载观测数据
x=w9497d;
dt=1; N=length(x); %由于数据的采样间隔为天
n=0:N-1;t=n*dt;f=n/(N*dt); %时间序列和频率序列
subplot(2,1,1),plot(n/365.25,x); %以年为单位给出记录数据随时间的变化,1 年有 365.25 天
xlabel('时间/年');
title('电导率数据');
```







```
ylabel('MS/CM');
y=fft(x);
f1=0.0;f2=0.005;                                %要滤去频率的上限和下限,即去掉频率范围为 0~0.005 天-1
yy=zeros(1,length(y));                            %设置与 y 相同元素的数组
for m=0:N-1                                         %将频率落在该频率范围及其大于奈奎斯特频率的波滤去
    if(m/(N*dt)>f1&m/(N*dt)<f2)...                 %小于奈奎斯特频率的滤波范围
        |(m/(N*dt)>(1/dt-f2)&m/(N*dt)<(1/dt-f1))    %大于奈奎斯特频率的滤波范围
        %1/dt 为一个频率周期
        yy(m+1)=0.;                                %置在此频率范围内的振动振幅为零
    else
        yy(m+1)=y(m+1);                            %其余频率范围的振动振幅不变
    end
end
subplot(2,1,2),plot(n/365.25,real(ifft(yy)))      %绘制滤波后的数据运用 IFFT 变换回时间域并绘图
title('滤波后的逆 Fourier 输出');
ylabel('MS/CM');
xlabel('时间/年')
```

程序运行结果如图 D-7 所示。可见采用这种滤波方法，有效地滤除了年变周期，使得可以更容易地分析信号其他频率的特征。

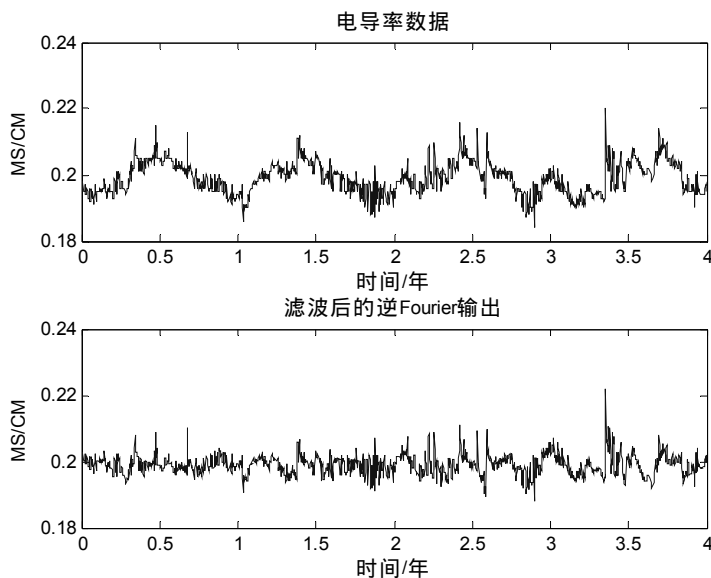


图 D-7 地电台数据滤除年变分量的结果



## D.3 FFT 在工程分析中的应用

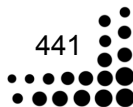
### 1. FFT 在地倾斜数据中的应用

作为一个例子，我们将某倾斜数字垂直摆的 2 个月 NS 向观测数据进行分析，由于数据为分钟值，所以采样周期为 1min。根据固体潮的理论可知，固体潮主要具有日潮、半日潮和三分之一潮。我们在频率域中分别绘出周期为 22 ~ 26h、11 ~ 13h 和 7.5 ~ 8.5h 的频率区间来分析该记录中是否含有这些频率的固体潮汐。

程序如下：

```
close all
load c818312b.dat;           %加载观测数据
dt=1;                         %采样间隔为分钟
x=c818312b;N=length(x);t=[0:N-1]*dt;
figure(1)                     %绘出记录数据随时间的变化
plot(t/(24*60),x);
%以天为单位给出原始记录，1 天为 24 × 60min
xlabel('时间/天');
title('地倾斜数据');
ylabel('振幅');
figure(2)
x=detrend(x);                 %去掉长期趋势（这里为渐增的趋势）
y=fft(x);                     %对该数据进行频谱分析
plot([0:N-1]/(N*dt),abs(y)*2/N); %绘出振幅谱
xlabel('频率/分钟^-1');
title('地倾斜数据的振幅谱');
ylabel('振幅');
xlim([0 1/(4*60)]);           %只绘出周期大于 4h（240min）的振幅
hold on;plot([1/(26*60) 1/(26*60)],ylim,'k:');
plot([1/(22*60) 1/(22*60)],ylim,'k:');
plot([1/(13*60) 1/(13*60)],ylim,'k:');
plot([1/(11*60) 1/(11*60)],ylim,'k:');
plot([1/(8.5*60) 1/(8.5*60)],ylim,'k:');
plot([1/(7.5*60) 1/(7.5*60)],ylim,'k:');
text(1/(24*60), 2, '日潮', 'rotation',90)
text(1/(12*60), 2, '半日潮', 'rotation',90)
text(1/(8*60), 2, '三分之一潮', 'rotation',90);
hold off
```

程序运行结果如图 D-8 和图 D-9 所示。从图 D-9 中可看出，倾斜数字观测数据明显含有日潮、半日潮和三分之一潮的信息，且半日潮的振幅最大，日潮次之，三分之一潮的振幅





最小，这与固体潮理论是一致的。但如果单从观测数据分析，则很难得出其中的优势频率成分。

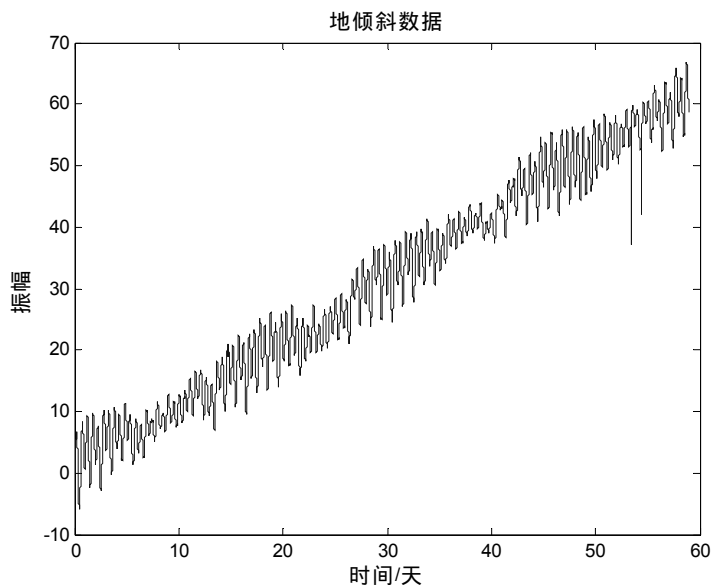


图 D-8 地倾斜观测数据

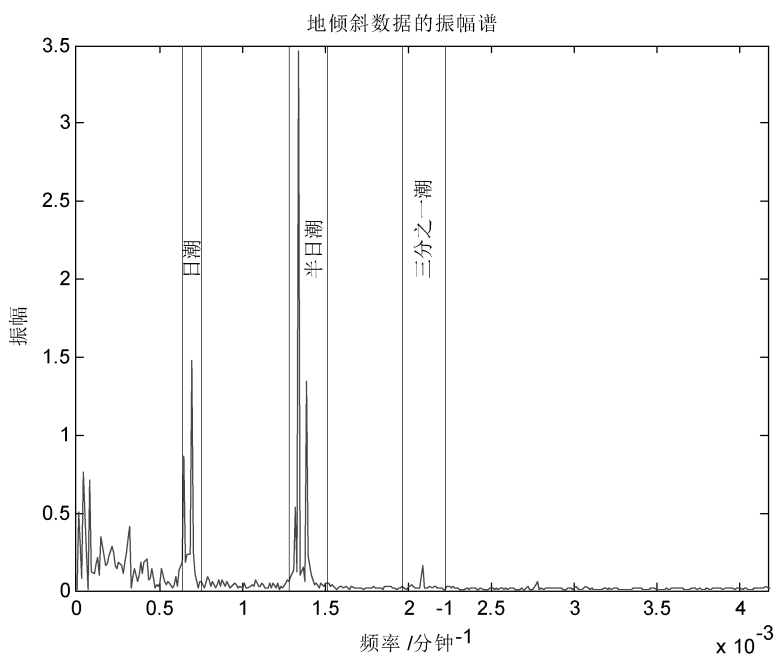


图 D-9 地倾斜观测数据的谱分析结果

## 2. 分析地震数据中的频率成分

地震数据中一般含有多种频率成分，有时我们需要分析地震波中含有哪种频率成分。作



为一个例子,这里处理某地震记录文件 2003012001653.vbb 的 has 台站的上下向记录,取该地震信号的文件 has.dat (P 波数据) \ has1.dat(S 波数据) \ has3.dat (面波数据) 来分析各种地震波的不同频率成分。对整个地震波信号、截取的 P 波信息、S 波信息和面波信号进行频率分析的程序如下:

```
load hns.dat ; %读取数据序列
Xt=hns; %把数据赋值给变量
Fs=50; %设定采样率(Hz)
dt=1/Fs; %求采样间隔(s)
N=length(Xt); %得到序列的长度
Xf=fft(Xt); %对信号进行快速 Fourier 变换(FFT)

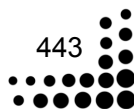
figure(1)
subplot(2,1,1),plot([0:N-1]/Fs,Xt); %绘制原始值序列
xlabel('时间/s'),title('时间域');
grid on;

subplot(2,1,2),plot([0:N-1]/(N*dt),abs(Xf)*2/N); %绘制信号的振幅谱
xlabel('频率/Hz'),title('幅频图');
ylabel('振幅');
xlim([0 2]); %频率轴只画出 2Hz 频率之前的部分
grid on;

figure(2)
load hns1.dat ; %读取数据序列
Xt=hns1; %把数据赋值给变量
Fs=50; %设定采样率(Hz)
dt=1/Fs; %求采样间隔(s)
N=length(Xt); %得到序列的长度
Xf=fft(Xt); %对信号进行快速 Fourier 变换(FFT)

subplot(2,1,1),plot([0:N-1]/Fs,Xt); %绘制原始值序列
xlabel('时间/s'),title('时间域');
grid on;

subplot(2,1,2),plot([0:N-1]/(N*dt),abs(Xf)*2/N); %绘制信号的振幅谱
xlabel('频率/Hz'),title('幅频图');
ylabel('振幅');
xlim([0 2]); %频率轴只画出 2Hz 频率之前的部分
grid on;
```





```
figure(3)
load hns2.dat ; %读取数据序列
Xt=hns2; %把数据赋值给变量
Fs=50; %设定采样率(Hz)
dt=1/Fs; %求采样间隔(s)
N=length(Xt); %得到序列的长度
Xf=fft(Xt); %对信号进行快速 Fourier 变换(FFT)

subplot(2,1,1),plot([0:N-1]/Fs,Xt); %绘制原始值序列
xlabel('时间/s'),title('时间域');
grid on;

subplot(2,1,2),plot([0:N-1]/(N*dt),abs(Xf)*2/N); %绘制信号的振幅谱
xlabel('频率/Hz'),title('幅频图');
ylabel('振幅');
xlim([0 2]); %频率轴只画出 2Hz 频率之前的部分
grid on;

figure(4)
load hns3.dat ; %读取数据序列
Xt=hns3; %把数据赋值给变量
Fs=50; %设定采样率(Hz)
dt=1/Fs; %求采样间隔(s)
N=length(Xt); %得到序列的长度
Xf=fft(Xt); %对信号进行快速 Fourier 变换(FFT)

subplot(2,1,1),plot([0:N-1]/Fs,Xt); %绘制原始值序列
xlabel('时间/s'),title('时间域');
grid on;

subplot(2,1,2),plot([0:N-1]/(N*dt),abs(Xf)*2/N); %绘制信号的振幅谱
xlabel('频率/Hz'),title('幅频图');
ylabel('振幅');
xlim([0 2]); %频率轴只画出 2Hz 频率之前的部分
grid on;
```

程序运行结果如图 D-10 ~ 图 D-13 所示。由图可见,经过快速 Fourier 变换 (FFT) 后可以看出不同地震震相的频谱分布特征是不一样的,P 波的频谱分布成分居多,最高到 1.2Hz 左右;S 波高频成分次之,频率成分主要分布在 0.2Hz 左右;面波的主要频率成分是低频 0.1 ~ 0.2Hz,得到了不同震相具有不同频率成分的结论。

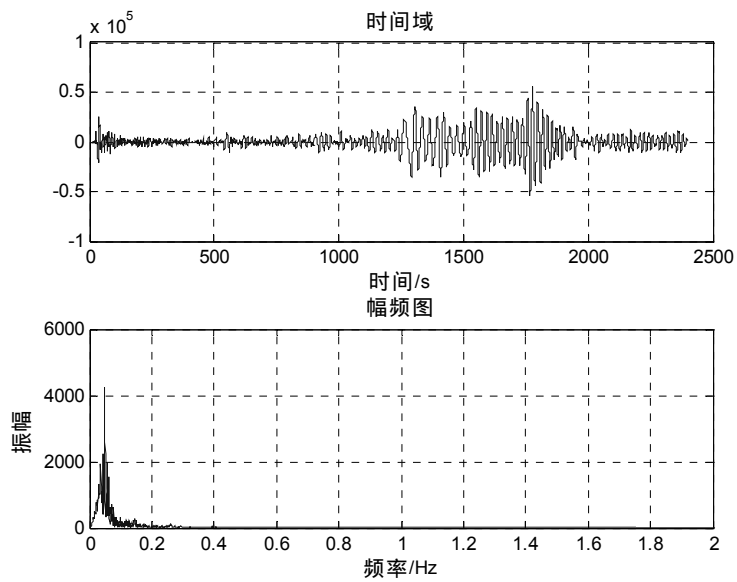


图 D-10 地震信号波形及波谱分析图

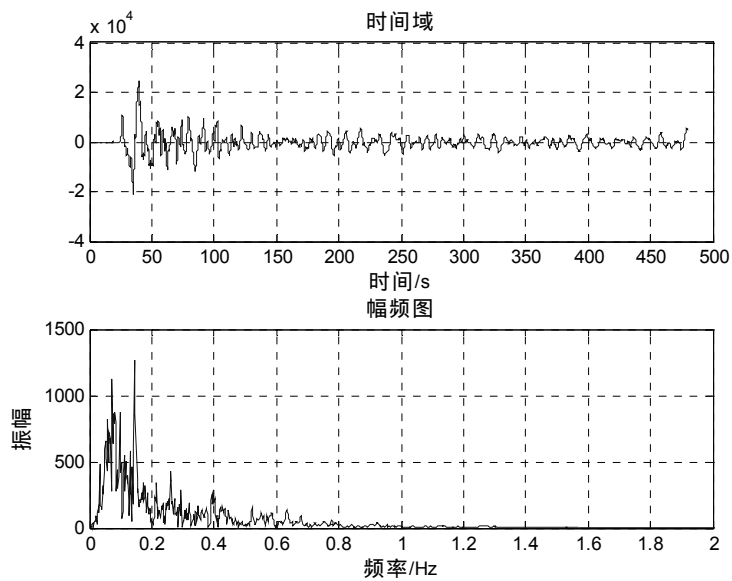


图 D-11 P 波信号波形及波谱分析图

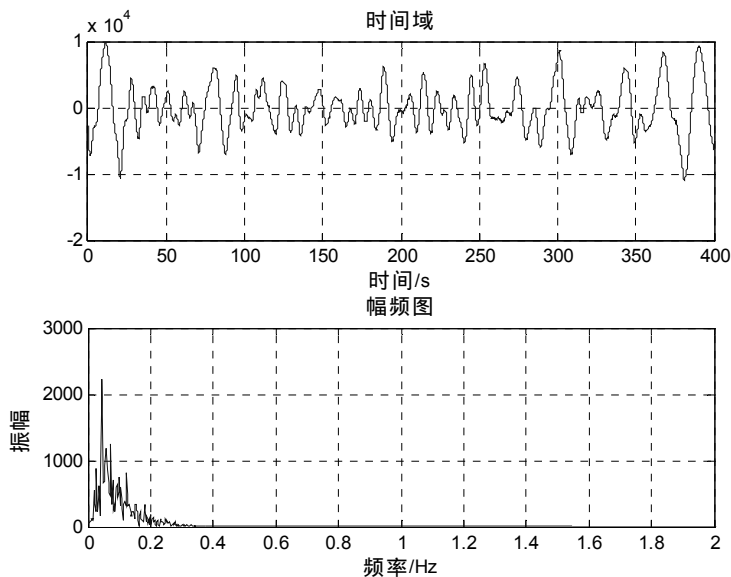


图 D-12 S 波信号波形及波谱分析图

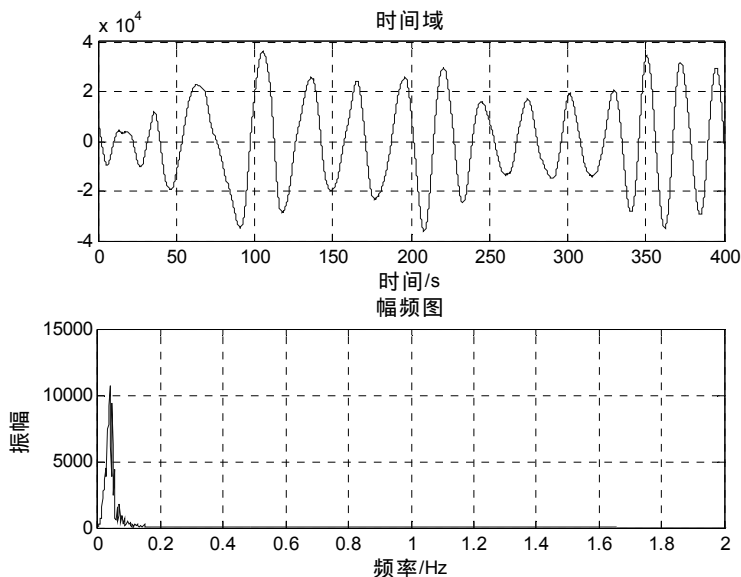


图 D-13 面波信号波形及波谱分析图

### 3. FFT 滤波的应用

前面我们给出了运用 FFT 进行简单滤波的尝试,这里将运用 FFT 对被高频和低频噪声干扰的地震信号进行滤波。我们选择某地震 yn9706300036.ev2 的 zds 台的记录。运用地震波截取软件得到其中垂直向地震记录文件为 zdsud.dat,运行下面程序,首先分析其频率成分:

```
%读入文本数据,绘制地震波时间域曲线及频谱分析图
```



```

load zdsud.dat; %读取数据序列
Xt=zdsud; %把数据赋值给变量
Fs=50; %设定采样率(Hz)
dt=1/Fs; %求采样间隔(s)
N=length(Xt); %得到序列的长度
Xf=fft(Xt); %对信号进行快速 Fourier 变换(FFT)
figure(1)
subplot(2,1,1),plot([0:N-1]/Fs,zdsud); %绘制原始值序列
xlabel('时间/s');
title('原始地震波记录');
grid on;
subplot(2,1,2),plot([0:N-1]/(N*dt),abs(Xf)*2/N); %绘制信号的振幅谱
xlabel('频率/Hz');
title('幅频图');
ylabel('振幅');
xlim([0 Fs/2]); %频率轴只画出奈奎斯特频率之前的部分
grid on;

```

得到的图形如图 D-14 所示。由图可见，其波形记录中含有高频和低频干扰，使得地震波形部分被噪声淹没。在幅频图中，可以看到有很多高频成分，并且前面的大幅度低频成分也可能是一种干扰。

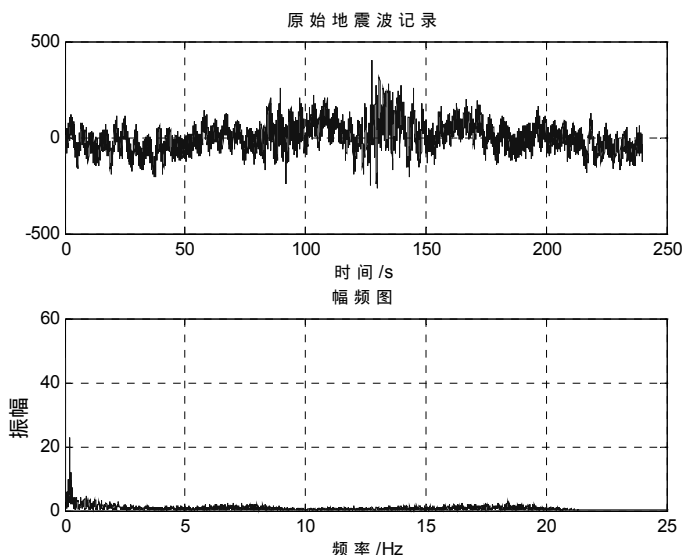


图 D-14 地震时域波形与幅频分析图

为了更清楚地显示地震信息，我们保留 0.6 ~ 1.8Hz 的频率成分而滤掉其他频率成分，程序如下：

```

load zdsud.dat; %读取数据序列
x=zdsud;

```







```
dt=0.02; N=length(x); %由于数据的采样间隔 0.02s
n=0:N-1;t=n*dt;f=n/(N*dt); %时间序列和频率序列
y=fft(x);
f1=0.6;f2=1.8; %要滤去频率的上限和下限
yy=zeros(size(y)); %设置与 y 相同元素的数组
for m=0:N-1 %将未在频率范围内的振动滤去
    if(m/(N*dt)>f1&m/(N*dt)<f2)... %小于奈奎斯特频率的保留频率范围
        |(m/(N*dt)>(1/dt-f2)&m/(N*dt)<(1/dt-f1)) %大于奈奎斯特频率的保留频率范围
        yy(m+1)= y(m+1); %保留频率范围内的振动振幅不变
    else %其余频率范围的振动振幅为零
        yy(m+1)=0.;
    end
end
subplot(2,1,1),plot(t,zdsud); %绘制原始值序列
xlabel('时间/s'),title('原始信号')
grid on;
subplot(2,1,2),plot(t,real(ifft(yy))) %绘制滤波后的数据运用 IFFT 变换回时间域并绘图
title('FFT 滤波后的信号图');
xlabel('时间/s')
grid on;
```

程序运行结果如图 D-15 所示。由图可见，滤波后可清楚地显示出地震的 P、S 震相。但在实际中，要多次尝试滤掉其中的干扰成分，以期达到最佳效果。

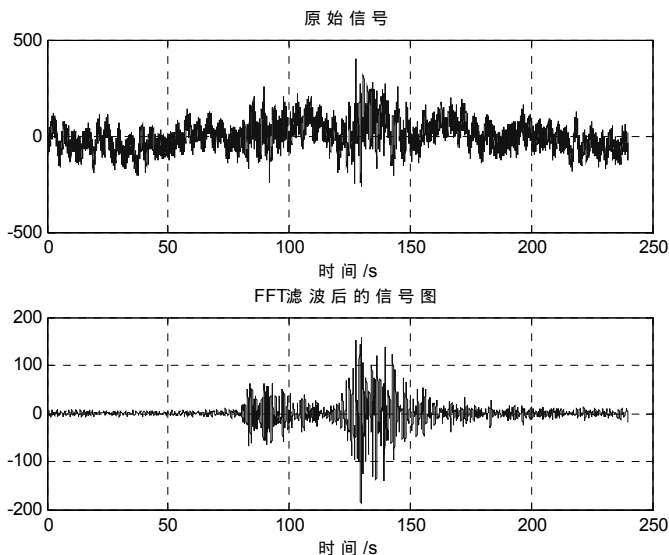


图 D-15 滤波前波形与滤波后的波形图

# 参 考 文 献

- [1] 孔玲军. MATLAB 小波分析超级学习手册[M]. 北京: 人民邮电出版社, 2014.
- [2] 张德丰. MATLAB 小波分析[M]. 北京: 机械工业出版社, 2008.
- [3] 高成, 董长虹, 郭磊, 等. MATLAB 小波分析与应用[M]. 2 版. 北京: 国防工业出版社, 2004.
- [4] 葛哲学, 陈仲生. MATLAB 时频分析技术及其应用[M]. 北京: 人民邮电出版社, 2005.
- [5] 飞思科技产品研发中心. MATLAB 6.5 辅助图像处理[M]. 北京: 电子工业出版社, 2003.
- [6] 刘正君. MATLAB 科学计算与可视化仿真[M]. 北京: 电子工业出版社, 2009.
- [7] 周伟, 桂林, 周林, 等. MATLAB 小波分析高级技术[M]. 西安: 西安电子科技大学出版社, 2005.
- [8] 葛超, 王蕾, 曹秀爽. MATLAB 技术大全[M]. 北京: 人民邮电出版社, 2014.
- [9] 葛哲学. 精通 MATLAB[M]. 北京: 电子工业出版社, 2008.
- [10] 张德丰. MATLAB 数值分析与应用[M]. 北京: 国防工业出版社, 2006.
- [11] 张强, 王正林. 精通 MATLAB 图像处理[M]. 2 版. 北京: 电子工业出版社, 2012.
- [12] 张德丰. 数字图像处理 (MATLAB 版) [M]. 北京: 人民邮电出版社, 2009.